VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Zpracování dat pro modulové řešení robota

# Data Processing for the Modular Robot Solution

2019 Bc. Rostislav Vondrák

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Zadání diplomové práce

Student: **Bc. Rostislav Vondrák**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Zpracování dat pro modulové řešení robota
Data Processing for the Modular Robot Solution

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem této práce je prostudovat využitelné metody a navrhnout řešení zpracování dat pro modulové řešení robota, vybrané metody implementovat na platformě Rapsberry pi.

1. Prostudujte problematiku zpracování a získávání signálů.
2. Vyberte vhodné metody pro zpracování a vyhodnocení signálů.
3. Implementujte vybrané algoritmy (po poradě s vedoucím) pro platformu Raspberry pi.
4. Experimentujte a ověřte navržené řešení a jeho spolehlivost.
5. Vhodně zvolte reprezentaci získaných výsledků a uvažte koncept využití navrženého řešení.

Seznam doporučené odborné literatury:

[1] Maragos P., Potamianos, A., Gros, P.: Multimodal Processing and Interaction: Audio, Video, Text (Multimedia Systems and Applications), Springer, ISBN-10: 1441945482, ISBN-13: 978-1441945488, 2010.
[2] Mohammed J. Zaki and Wagner Meira, Jr., Data Mining and Analysis: Fundamental Concepts and Algorithms, Cambridge University Press, May 2014. ISBN: 9780521766333.
[3] Robert M. Gray and Lee Davisson, An Introduction to Statistical Signal Processing, Cambridge University Press, 2004.
[4] Spyrou, E., Iakovidis, D. and Mylonas P.: Semantic Multimedia Analysis and Processing (Digital Imaging and Computer Vision), CRC Press, ISBN-10: 1466575492, ISBN-13: 978-1466575493, 2014.
[5] https://www.raspberrypi.org/documentation.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Jana Nowaková, Ph.D.**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
*vedoucí katedry*

prof. Ing. Pavel Brandštetter, CSc.
*děkan fakulty*

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2019 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 30. dubna 2019 .....

Chtěl bych poděkovat své vedoucí práce Ing. Janě Nowakové, Ph.D. za vedení, rady a čas, který mi věnovala během tvorby práce. Rád bych rovněž poděkoval mé rodině a blízkým přátelům za pomoc a podporu během studia.

**Abstrakt**

Předložená diplomová práce se zabývá zpracováním signálů a popisem dostupných metod určených k jejich analýze. Je zde popsána problematika práce se zvukovým a obrazovým signálem zahrnující jeho získání, zpracování a následné vyhodnocení informací. U obrazového signálu se provádí detekce obličeje v obraze a následné rozpoznání pohlaví z tohoto obličeje. Analýza zvukového signálu je zaměřena na určení směru zvuku pomocí mikrofonového pole. Výsledkem práce je pohyblivý robot využívající popsané metody schopný provádět jednoduché úkony na základě vysloveného příkazu.

**Klíčová slova**: analýza signálů, obrazový signál, zvukový signál, detekce obličeje, rozpoznání pohlaví, určení směru zvuku, korelace signálů, digitalizace, redukce dat, Rapsberry Pi

**Abstract**

This thesis deals with data processing and description of available methods used for their analysis. The main focus is on processing audio and video signals which includes signal acquisition, processing and subsequent evaluation of extracted information. The image analysis includes face detection in the image followed by gender recognition from that face. The audio signal analysis is aimed at determining the direction of the sound using the microphone array. The result of this work is a robot which uses described methods capable of performing simple tasks based on a spoken command.

**Key Words**: signal analysis, video signal, audio signal, face detection, gender recognition, sound direction determination, signal correlation, digitalization, data reduction, Raspberry Pi

# Obsah

# Seznam použitých zkratek a symbolů

| | | |
|---|---|---|
| 3D | – | Trojdimenzionální |
| AD | – | Analogově digitální |
| API | – | Application Programming Interface |
| DSI | – | Camera Serial Interface |
| COM | – | Communication port |
| CSV | – | Comma-Separated Values |
| DFT | – | Discrete Fourier Transform |
| DSI | – | Display Serial Interface |
| FFS | – | Fusion of Facial Strips |
| FFT | – | Fast Fourier Transform |
| FP | – | False Positive |
| FN | – | False Negative |
| HAT | – | Hardware on Top |
| HDMI | – | High-Definition Multi-media Interface |
| HW | – | Hardware |
| I2C | – | Inter-Integrated Circuit |
| IDE | – | Integrated Development Environment |
| IoT | – | Internet of Things |
| GPIO | – | General Purpose Input/Output |
| LAN | – | Local Area Network |
| LBP | – | Local Binary Pattern |
| LBPH | – | Local Binary Pattern Histogram |
| LDA | – | Linear Discriminant Analysis |
| LED | – | Light Emitting Diode |
| MEMS | – | MicroElectroMechanical System |
| NFC | – | Near Field Communication |
| OS | – | Operační systém |
| PC | – | Personal Computer |
| PCA | – | Principal Component Analysis |
| PMDL | – | Personal model file type |
| PWM | – | Pulse-Width Modulation |
| PGM | – | Portable Gray Map |
| SPI | – | Serial Peripheral Interface |
| SoC | – | System on a Chip |
| SSH | – | Secure Shell |
| SVM | – | Support Vector Machines |

| | | |
|---|---|---|
| TP | – | True Positive |
| TTS | – | Text To Speech |
| UMDL | – | Universal model file type |
| USB | – | Universal Serial Bus |
| VS | – | Visual Studio |
| WAV | – | Waveform audio file format |
| XML | – | Extensible Markup Language |

# Seznam obrázků

# Seznam tabulek

# 1 Úvod

Zpracování dat je proces umožňující získat informace z různě reprezentovaných dat. Předložená práce se věnuje především získání signálů různých fyzikálních veličin, převodu těchto signálů do srozumitelné formy pro výpočetní techniku a jejich následnou analýzu a praktickou demonstraci využití.

Cílem práce je tedy seznámení se s existujícími postupy a algoritmy pro zpracování audio a video signálů, implementace vybraných algoritmů, jejich ověření a praktická demostrace jejich využití v rámci fyzické realizace robota.

Práce je rozdělena do dvou hlavních částí, a to teoterické části a části praktické, která demonstruje využití vybraných metod pro zpracování informací získaných z reálného světa. Teoretická část je zaměřena na zpracování audio signálů získaných pomocí mikrofonového pole a video signálů snímaných kamerou. U audio signálů se v rámci řešení práce jedná o určení směru zdroje zvuku vůči mikrofonovému poli v horizontální rovině tedy umísění zdroje zvukového signálu a rozpoznání klíčového slova v mluveném textu. Analýza video signálu je zaměřena na identifikaci obličeje v obraze a rozpoznání pohlaví snímané osoby z tohoto obličeje. Pro člověka jsou oba tyto úkoly zcela přirozené a děláme je běžně, ale v číslicové technice se jedná o komplexní analýzy zahrnující spoustu kroků jejichž správné provedení ovlivňuje kvalitu získaných výsledků.

První částí praktické části práce je fyzická realizace robota, který by měl být složen z komponent umožňující provedení jednotlivých analýz z teoretické části. Výběr komponent je velmi důležitý, protože ovlivňuje kvalitu získáných dat, které se následně zpracovávají. Druhou částí praktické realizace je implementace programu, který využije metody popsané v teoretické části. Očekávaným výsledkem je, aby robot korektně zpracovával získané informace a dokázal výsledky interpretovat uživateli. Využitou platformou určenou pro zpracování dat je Raspberry Pi, které s využitím přídavného výpočetního hardwaru a doplňkových senzorů dokáže získávat data z okolního světa a následně je zpracovat pomocí popsaných metod. Zároveň by měla být umožněna maximální mobilita robota po fyzické stránce a rovněž by neměl být závislý na prostředí, ve kterém se nachází. Z těchto důvodů by měl disponovat vlastním zdrojem energie, který dokáže napájet všechny moduly robota a rovněž by neměl disponovat kabeláží, která by ho jakkoliv omezovala v pohybu. Pro zachování mobility bez nutnosti dodatečné konfigurace před spuštěním robota by všechny implementované algoritmy měly být nezávislé na připojení k internetu.

V závěru práce je demonstrováno využití a ověření funčnosti navrženého řešení v podobě měření účinnosti jednotlivých popsaných metod pro vlastní a volně dostupná data.

# 2 Analýza a zpracování audio signálů

## 2.1 Získání, konverze a předzpracování signálu

Veškerá analýza zvukového signálu v této práci je zaměřena pouze na lidský hlas. Ten je, ale nutné nejprve převést do digitální formy. Cílem této kapitoly je tedy popsat převod hlasu do digitální formy vhodné k analýze.

Lidské hlasivky tvoří zvukový signál kmitáním a tím způsobují rychlé změny tlaku vzduchu a kmitání molekul vzduchu. Toto kmitání vytváří vlnění, které člověk vnímá ušima. Proces zachycení tohoto vlnění a jeho zpracování se ve výpočetní technice musí nahradit kombinací zařízení a algoritmů.

Pro zachycení zvukového signálu se používají mikrofony, které v kombinaci se zesilovači dokáží převést zvukový signál na elektrickou veličinu. S využitím výpočetní techniky jsme schopni takový signál zpracovávat a analyzovat.

### 2.1.1 Digitalizace

Digitalizací signálu se rozumí převod z analogové podoby do digitální. Signál o nekonečném počtu úrovní je převeden na signál s konečným počtem a dochází tím ke ztrátě informace. Proces digitalizace signálu je složen z několika kroků, které ovlivňují podíl ztráty informace a tím i výslednou kvalitu převodu.

**2.1.1.1 Vzorkování** Vzorkování je část procesu digitalizace, v němž se přiřazují spojitému signálu v daných časových okamžicích určité funkční hodnoty. Tyto časové okamžiky jsou určeny vzorkovací frekvencí. Signál je tedy navzorkován podle vybrané vzorkovací frekvence Diracovými impulzy a funkční hodnoty signálu v momentech těchto impulzů představují výsledné hodnoty navzorkovaného signálu. [1]

Vzorkovací frekvenci je potřeba zvolit, tak aby nedocházelo k výraznému zkreslení při ztrátě informace. Tento problém popisuje a řeší Shannon-Kotělníkův teorém o vzorkovací frekvenci: *„Přesná rekonstrukce spojitého, frekvenčně omezeného signálu z jeho vzorků je možná tehdy, pokud byla vzorkovací frekvence vyšší než dvojnásobek nejvyšší harmonické složky vzorkovaného signálu.“* [2]

V praxi to znamená, že vzorkovací frekvenci je potřeba zvolit dvakrát větší než je maximální požadovaná přenášená frekvence signálem. K tomuto se ve většině případů přidává navíc i rezerva 10%. Frekvenční rozsah lidského sluchu je od 16Hz do 20kHz, tudíž při aplikaci teorému získáváme hodnotu 44kHz vzorkovací frekvence. [1]

**2.1.1.2 Kvantování** Při procesu kvantování se přiřazuje navzorkovanému spojitému signálu přiřadí hodnota určité veličiny. V tomto případě se jedná o napětí na vstupu do zvukové karty. Tento převod hodnot analogového signálu na digitální zajišťuje AD převodník.

Digitální signál představuje výhodu v menším objemu dat a usnadnění při následném zpracování. Nevýhodou je pak nenávratná ztráta informací, kterou obsahoval původní spojitý signál. Z tohoto důvodu je nutné vybrat správné vlastnosti AD převodníku v závislosti na parametrech vstupního signálu. [4]

- Rozlišovací schopnost – udává kolika různých hodnot může signál v jednotlivých chvílích nabývat. Udává se v bitech, pro běžné použití stačí 16bitové rozlišení. Profesionální řešení nabízí i 24 bitů.
- Kvantovací krok – jedná se o rozdíl v hodnotách napětí analogového signálu, který určuje přechod z jedné digitální hodnoty na jinou.
- Chyba kvantování – udává maximální hodnotu rozdílu mezi hodnotou analogového signálu a digitálního ve stejný okamžik.

**2.1.1.3 Normování** Pro usnadnění práce a zpracování signálu se hodnoty signálu normují podle následujícího vztahu

$$w(n) = \frac{w(n)}{MAX\,|w(n)|}, \tag{1}$$

kde $n$ je index vzorku. Hodnota se upraví podle nejvyšší hodnoty amplitudy a tím se výsledné hodnoty dostanou do intervalu $\langle -1; 1 \rangle$. To je umožněno hlavně z důvodu toho, že pro zpracování nás nezajímají reálné hodnoty signálu, ale pouze poměr vůči největší hodnotě. [1]

### 2.1.2 Segmentace

Segmentace signálu je krok ve zpracování, při kterém se dělí signál na menší části. Tyto části jsou stejně dlouhé a dohromady skládájí původní signál. Segmentace je důležitá pro následnou analýzu, jelikož přímo ovlivňuje počet segmentů, které budou zpracovávány. Při volbě velikosti segmentu je důležité myslet na vzorkovací teorém, ale rovněž je u některých analýz nutné mít dostatečný počet segmentů. Počet segmentů představuje podíl mezi celkovou délkou audio signálu a délkou jednoho segmentu. [1]

## 2.2 Zpracování zvukových dat

Po provedení všech kroků v předchozí kapitole o předzpracování audio signálu získáme digitální audio data rozdělené do segmentů a připravené ke zpracování. Zpracování zahrnuje aplikaci několika algoritmů, díky nimž bude možné rozpoznávat směr odkud přichází zvuk k robotovi. [1]

## 2.3 Rozpoznání směru zvuku

V implementační části projektu je využito mikrofonové pole nejen jako vstupní zařízení pro rozpoznání klíčových slov, ale díky rozmístění jednotlivých mikrofonů do kruhu je použito rovněž k lokalizaci směru zvukového signálu.

Lokalizace zdroje zvuku je v rámci projektu potřebná pouze ve 2D rovině horizontálně. Metody rozpoznání směru můžou pracovat s různými proměnnými. Nejčastějším rozhodujícím prvkem je čas, ale rovněž se dá pracovat s intenzitou (hlasitostí) zvuku. [3]

### 2.3.1 Vzájemná korelace

Vzájemná korelace je metoda porovnávání signálů, která se zaměřuje na jejich podobnost. U audio signálů se porovnává jestli je zvuk nahraný ze dvou mikrofonů podobný a případně se dá rovněž zjistit posunutí těchto signálů mezi sebou.

Obecně se vzájemná korelace signálů dvou diskrétních signálů x a y o délce N určuje podle následujícího vztahu

$$Corr\left\{x,y\right\}\left[n\right]=\sum_{n=1}^{N}x\left[m\right]\cdot y\left[m+n\right]. \tag{2}$$

Z tohoto vztahu je zřejmé, že porovnání probíhá tak, že se jeden signál posunuje vůči tomu druhému. Blíže popsáno v kapitole 2.3.1.2.

Zvuky se kterými se denně setkáváme nejsou složeny pouze z jedné frekvence, ale většinou mají složitý průběh. Signál, který získáme z mikrofonů je výsledkem konvoluce signálů určitých frekvencí a představuje tedy jejich kombinaci. Pro zjištění těchto frekvencí potřebujeme získat spektrum zvukového signálu. K tomu slouží Fourierova transformace. [6]

**2.3.1.1 Fourierova transformace** Fourierova transformace se obecně používá k převodu časové složky signálu na frekvenční. Používá se v případě, kdy chceme určit spektrum spojitého konečného signálu. V této práci se zabývám pouze analýzou digitálních signálů, takže budu používat Diskrétní Fourierovu transformace (DFT) a její modifikaci rychlou Fourierovou transformací (FFT).

Pro využití DFT je zapotřebí mít signál, který je diskrétní jak v čase tak i v frekvenční oblasti. Signál má tedy stejný konečný počet hodnot v obou případech. Jejím cílem je spočítat koeficienty Fourierových řad navzorkovaného signálu. Je to výpočetně složitý algoritmus obsahující $(N-1)^2$ komplexních součinů a $N\left(N-1\right)$ komplexních součtů.

Výpočet DFT je definován jako

$$X\left[k\right]=\frac{1}{N}]\sum_{n=0}^{N-1}x\left[n\right]e^{-i\frac{2n}{N}kn}. \tag{3}$$

kde $x\left[n\right[\in\ C$ (obor komplexních čísel).

Exponenciální člen $e^{-i\frac{2n}{N}kn} = cos\left(\frac{2n}{N}kn\right) - isin\left(\frac{2n}{N}kn\right)$ nahradíme $w_N^{kn}$ a získáme

$$X[k] = \frac{1}{N}] \sum_{n=0}^{N-1} x[n] e^{-i\frac{2n}{N}kn} w_N^{kn}. \tag{4}$$

Při využití obou složek komplexního čísla získáme matici $W_N[n,k] = w_N^{kn}$, kde n jsou indexy řádků matice a k jsou indexy sloupců.

$$W_N = \begin{bmatrix} w_N^{0\cdot 0} & w_N^{0\cdot 1} & w_N^{0\cdot 2} & \cdots & w_N^{0\cdot(N-1)} \\ w_N^{1\cdot 0} & w_N^{1\cdot 1} & w_N^{1\cdot 2} & \cdots & w_N^{1\cdot(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_N^{(N-1)\cdot 0} & w_N^{(N-1)\cdot 1} & w_N^{(N-1)\cdot 2} & \cdots & w_N^{(N-1)\cdot(N-1)} \end{bmatrix}$$

Což se dá zapsat jako

$$X = \frac{1}{N} W_N x. \tag{5}$$

Z důvodu velké časové náročnosti výpočtu DFT vznikl algoritmus FFT. Využívá vlastností diskétního signálu a dělí jej na dvě poloviny. Existuje několik variant algoritmu FFT. Nejčastěji používaným z nich je Cooley-Turkey. Rozdělení na poloviny v tomto algoritmu je provedeno rozdělením na sudou a lichou posloupnost. Na každé půlce signálu je následně proveden výpočet DFT. Aby mohlo dojít k postupnému dělení signálu na poloviny musí být délka vstupního signálu $N = 2^k$, kde $k \in N$ je počet vstupních vzorků.

Prvním krokem výpočtu je rozdělení na sudé a liché vzorky signálu

$$u[n] = x[2n] \quad a \quad v[n] = x[2n+1]. \tag{6}$$

Následně se při určování DFT pro jednotlivé vzorky z obou posloupností skládají dvoubodové DFT vzorky, poté čtyřbodové atd. Dokud nedostaneme DFT o původním počtu vzorků podle Obr. 1.

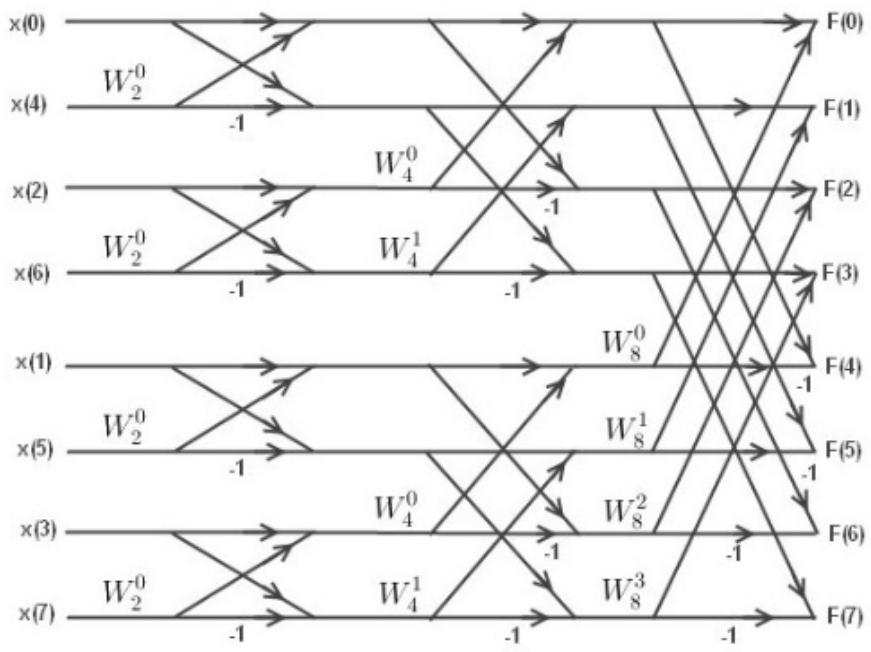
Při každém „skládání“ DFT vzorků se pracuje s dvěmi komplexními čísly $a$ a $b$. Číslo $b$ je vynásobeno hodnotou exponenciálního členu $w_N^{kn}$ a poté je přičteno nebo odečteno od čísla $a$. Výsledné čísla $A$ a $B$ jsou dále použity v průchodu algoritmem.

Při jednotlivých výpočtech je proveden jen jeden komplexní součin a dva komplexní součty. Díky tomu FFT snižuje počet výpočtů komplexních součinů na $(N^2)/2$.

[1], [5], [6]

**2.3.1.2 Výpočet korelace a určení směru** S využitím rychlé Fourierovy transformace jsme schopni porovnat podobnost dvou zvukových signálů z mikrofonů. Pro výpočet směru odkud jde zvuk se postupuje následovně

1. Mějme dva diskrétní signály $a$ a $b$ obsahující digitální audio signál o délce $n$.




Obrázek 1: Schéma výpočtu FFT pro 8 vzorků [7]

2. Pro oba signály se spočítá rychlá Fourierova transformace.

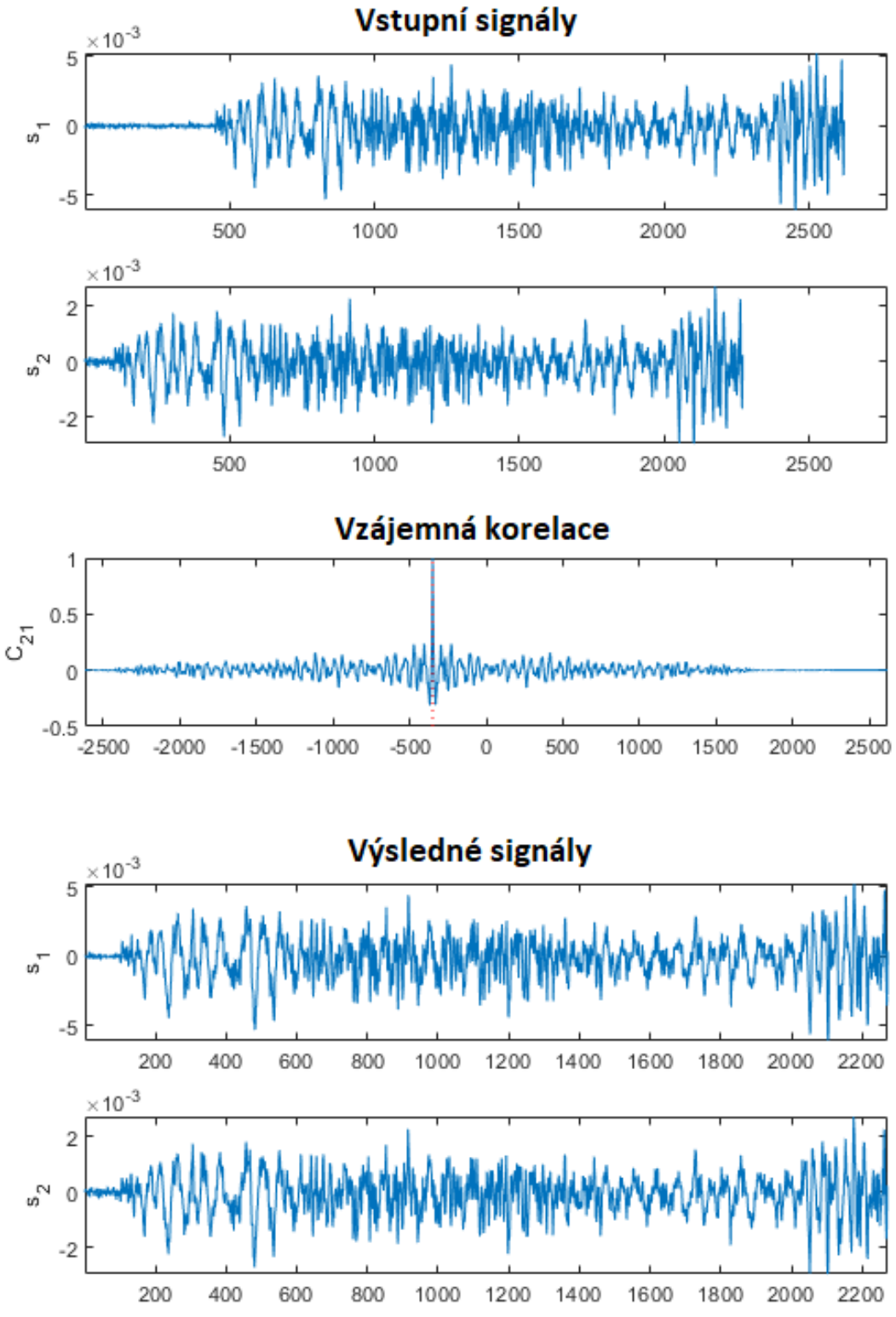
3. Provede se korelace transformovanych signalu $A'$ a $B'$. Jednotlivé hodnoty signálu $A'$ se vynásobí komplexně sdruženými hodnotami signálu $B'$ a vznikne výsledný signál $C'$.

4. Aplikujeme inverzní Fourierovu transformaci na signál $C'$ pro získání výsledku korelace

5. Ve výsledném signalu $C$ jsou hodnoty korelace signalu $A$ a $B$. Amplituda funkční hodnoty signalu $C$ představuje okamžik maximální podobnosti signálu $A$ a $B$. Hodnota ve které je amplituda funkce označuje hodnotu posunu mezi signály $A$ a $B$, viz ilustrace na Obr. 2.

Opakujeme-li tento postup pro všechny dvojice protějších mikrofonů v mikrofonovém poli dostaneme hodnoty posunu signálu všech čtyř dvojic mikrofonů. Poté najdeme dvojici s minimálním posunem, což signalizuje, že stejný zvuk dorazil k této dvojici mikrofonů ve stejný okamžik. Z toho můžeme předpokládat, že zdroj zvuku je této dvojici kolmý.

Tímto jsme omezili směr zdroje zvuku na dva směry. Pro identifikaci jednoho z nich stačí aplikovat korelační algoritmus pro dvojici mikrofonů, které jsou rovnoběžně se zdrojem zvuku resp. jsou kolmo k první dvojici mikrofonů. Výsledkem je určení jednoho z osmi možných směrů s tolerancí +- 22,5°. [3]




Obrázek 2: Aplikace vzájemné korelace na dva signály posunuté v čase [8]

# 3 Analýza a zpracování obrazu

V této kapitole budou popsány kroky potřebné k získání obrazu a jeho analýze. Popíšu několik druhů metod využitých v práci zaměřených na detekci obličeje v obraze a rozpoznání pohlaví konkrétních osob.

## 3.1 Zpracování obrazu

Proces zpracování začíná u získání obrazu. To je dnes možné použitím několika různých zařízení, z nichž nejběžnější jsou fotoaparáty, kamery a mobilní telefony. Snímače v nich fungují na stejném principu a slouží k zachycení obrazu ve spektru, které dokáže zachytit lidské oko. Následné kapitoly analýzy obrazu budou zaměřeny právě na toto pro člověka viditelné spektrum.

### 3.1.1 Digitalizace

Pro zpracování obrazu pomocí výpočetní techniky je nejprve nutné převést jeho fyzické (optické) vlastnosti na vlastnosti elektrické. Tento postup je složen z několika kroků, které jsou podobné těm při zpracování zvukového signálu

1. vzorkování – proces kdy dochází k prostorovému rozlišení (rozdělení na pixely),
2. kvantování – rozlišení a rozdělení jasových úrovní, které provádí převedení jasové složky do celočíselných hodnot.

Při digitalizaci obrazu se rovněž musí dbát na správnost provedení jednotlivých kroků, protože při nich dochází ke ztrátě informace, která je nevratná. Čím jemnější je vzorkování a kvantování, tím lépe je aproximován původní spojitý obrazový signál. Míru ztráty informace můžeme ovlivnit zvolením vhodných hodnot následujících parametrů.

- Vzorkovací interval – popisuje prostorové rozmezí vzorku v obraze a je přímo úměrné počtu vzorků. Stejně jako u zvukového signálu i zde se uplatňuje Shannon-Kotělnikův teorém [2].
- Vzorkovací mřížka – definuje plošné uspořádání vzorků, nejčastěji se používá tvar čtverce.
- Kvantovací úroveň – popisuje převod hodnot navzorkovaného signálu do množiny konečných hodnot. Hodnota každého vzorku představuje jas dané spektrální složky a ta je rozdělena do $k$ intervalů. Při výběru hodnoty se přihlíží k tomu, že lidské oko rozlišuje přibližně padesát jasových úrovní.

Jednobarevný digitalizovaný obraz je popsán maticí vzorků obsahujících hodnoty jasu dané barvy. U barevného obrazu je rozdíl, že hodnoty jsou nahrazeny vektory obsahující hodnoty jednotlivých složek barevného modelu. S touto datovou reprezentací obrazu se dá jednoduše pracovat a monochromatický obraz bude představovat základní formát který budu užívat v analýze. [9]

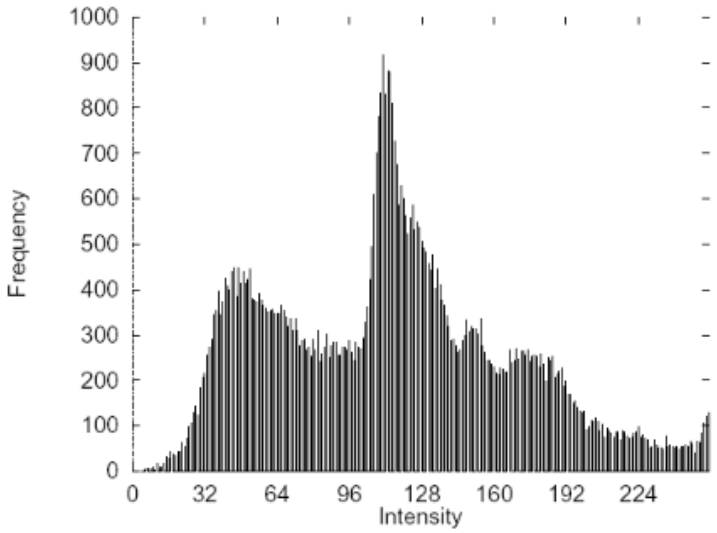
### 3.1.2 Korekce obrazu

Obraz je nyní reprezentován maticí vzorků (resp. vektorů) a je tedy digitalizován. Výsledek digitalizace nemusí být vždy uspokojivý a tím může ovlivnit výsledky následné analýzy. Proto je potřeba zjistit prvotní informace o digitálním obrazu pomocí histogramu.

Histogram udává informaci o rozložení jednotlivých úrovní jasových složek v obraze. Je reprezentován diskrétní funkcí

$$h\left(r_k\right) = n_k, \tag{7}$$

kde $r_k$ je k-tá úroveň šedi a $n_k$ je počet pixelů v obraze s úrovní šedi $r_k$.



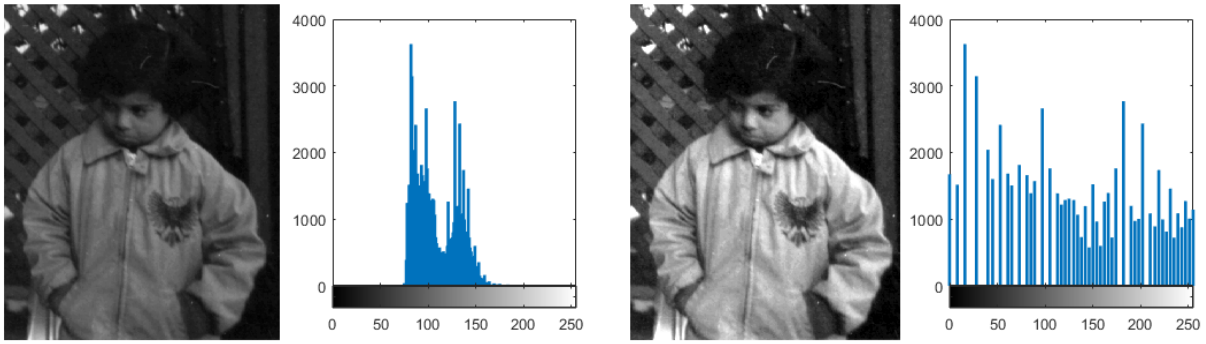

Obrázek 3: Histogram pro 256 jasových úrovní [9]

Reprezentace obrazu histogramem pomáhá při nastavování parametrů při získání obrazu a digitalizaci, případně při změnách jasové stupnice. Korekce jasové stupnice se provádí při ekvalizaci histogramu a je to hlavní krok při korekci obrazu.

Ekvalizace histogramu představuje vyrovnání histogramu. Je to transformace hodnot jasu při které se využijí všechny jasové úrovně a dojde tím k dosažení maximálního kontrastu mezi jednotlivými hodnotami. Zvýšením kontrastu se viditelně zvýrazní přechody mezi jasovými složkami a tyto přechody se nazývají „hrany“. [9]

## 3.2 Lokalizace obličeje v obraze – Viola-Jones

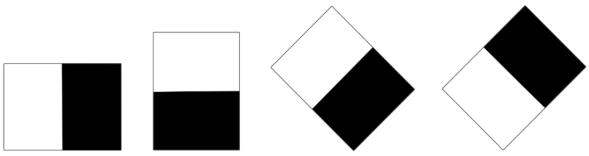
Pro nalezení obličeje v monochromatickém obraze se využívá změn v jasové složce obrazu. Zvolenou metodou pro lokalizaci je metoda využívající Haarovy příznaky popsaná v práci od Paul Viola a Michael Jones [10] vydané v roce 2001. Je to metoda strojového učení, která je založena na trénování na rozsáhlém datasetu pozitivních a negativních identifikací daného objektu. [10]



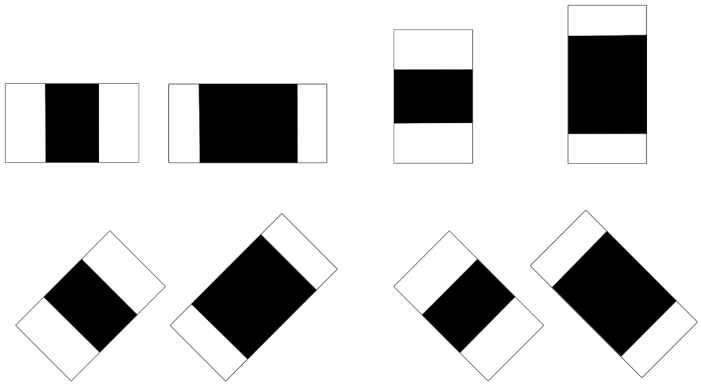

Obrázek 4: Výsledek ekvalizace histogramu [9]

### 3.2.1 Haarovy příznaky

Haarovy příznaky jsou vytvořeny na základě Haarových vlnek, což jsou jednotné čtvercové vlny s jedním vysokým a jedním nízkým intervalem. Je prezentována jako dvojice obdélníků, kde jeden je světlý a druhý tmavý. Kombinací těchto vlnek vznikají Haarovy příznaky. Základní sada příznaků se skládá ze tří druhů – hranové, čárové a středové příznaky.

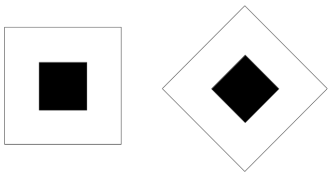


Obrázek 5: Hranové Haarovy příznaky [12]



Obrázek 6: Čárové Haarovy příznaky [12]

Každý příznak je reprezentován jako rozdíl mezi součtem pixelů pod bílou plochou a součtem pixelů pod černou plochou. Při generování příznaků jsou zvoleny výchozí hodnoty příznaku a poté je příznak postupně vertikálně nebo horizontálně posouván obrazem po jednom pixelu. Při



Obrázek 7: Středové Haarovy příznaky [12]

Tabulka 1: Druhy a počty Haarových příznaků pro velikost obr. 24*24 [12]

| Typ příznaku | Počet příznaků |
|---|---|
| hranový – 1A | 43 200 |
| hranový – 1A | 43 200 |
| čárový – 2A | 27 600 |
| čárový – 2B | 19 800 |
| čárový – 2C | 27 600 |
| čárový – 2D | 19 800 |
| středový – 3A | 8 464 |

každém posunu je příznak uložen do kolekce příznaků. Pokud dojde k pokrytí celého obrazu, je zvětšena jeho velikost a proces posouvání se opakuje. Výpočet probíhá dokud je velikost příznaku menší než velikost obrazu. Obrázek o velikosti 24*24 pixelů obsahuje přes 190 000 příznaků. [10]

**3.2.1.1 Integrální obraz** Pro efektivnější detekci Haarových příznaků v obraze je v metodě využit integrální obraz. Je to obraz o stejné velikosti jako originální, ale místo hodnot svítivosti na pozicích je v každém pixelu uložena hodnota součtu pixelů nad a nalevo od této aktuální pozice. Výpočet se řídí následujícím vztahem

$$ii\left(x,y\right)=\sum_{x'\leq x,y'\leq y}i\left(x',y'\right), \tag{8}$$

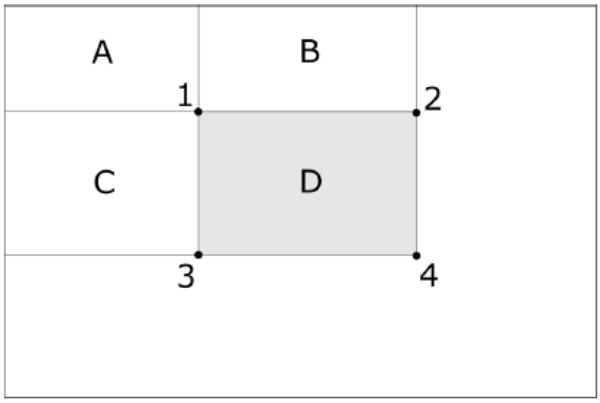
kde $ii\left(x,y\right)$ je integrální obraz a $i\left(x,y\right)$ je původní obraz. Pomocí vztahů

$$s\left(x,y\right)=s\left(x,y-1\right)+i\left(x,y\right), \tag{9}$$

$$ii\left(x,y\right)=ii\left(x-1,y\right)+s\left(x,y\right), \tag{10}$$

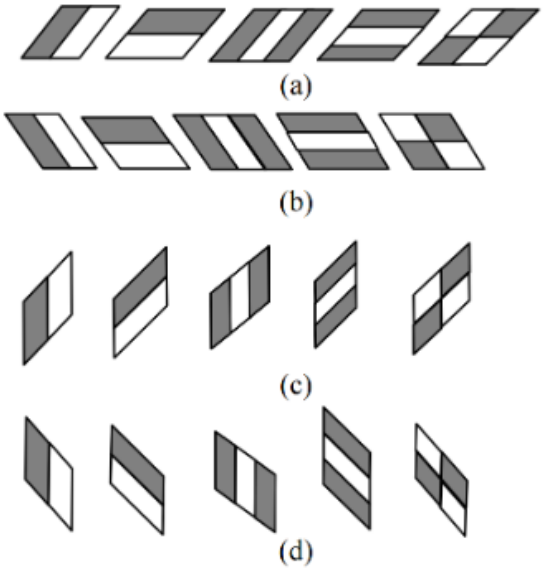
kde $s\left(x,y\right)$ je kumulovaný součet jasových hodnot v řádku a kde $s\left(x,-1\right)=0$ a $ii\left(-1,y\right)=0$ lze integrální obraz vypočítat při jednom průchodu původním obrazem.

Hlavní výhodou integrálního obrazu je, že není nutné počítat součet pixelů v oblastech v obraze procházením celého obrazu znovu. Pokud bychom chtěli znát součet pixelů v obrázku Obr. 8 v oblasti $D$, tak nám stačí znát hodnoty v krajních bodech 1,2,3,4 této oblasti. [10]




Obrázek 8: Příklad výpočtu integrálního obrazu obdélníkové plochy [12]

**3.2.1.2 Rozšiřující Haarovy příznaky** Hlavním nedostatkem sady Haarových příznaků je jejich omezení na výpočet pouze obdelníkových částí obrazu, což má za následek omezené možnosti detekce nepravidelných objektů. Z tohoto důvodu byly přidány rozšiřující Haarovy příznaky, které rozšiřují původní sadu příznaků a vylepšují samotnou detekci. Princip výpočtu těchto příznaků je stejný, nicméně je trochu složitější kvůli nutnosti výpočtu lichoběžníků. [11]




Obrázek 9: Rozšiřující sada Haarových příznaků [11]

### 3.2.2 AdaBoost

Vzhledem k velkému počtu příznaků, které se dají nalézt v malém obrázku by bylo časově náročné procházet všechny příznaky v dnešních obrázcích, které mívají několik miliónů pixelů. Navíc pouze část detekovaných příznaků má nějakou vypovídající hodnotu pro detekci. Proto se

využívá metoda strojového učení AdaBoost neboli Adaptive Boosting, která pomáhá s výběrem vhodných Haarových příznaků. AdaBoost kombinuje několik slabých klasifikátorů pro vytvoření silného klasifikátoru. Slabé klasifikátory mají výhodu, že jsou velmi přesné, ale je jich mnoho. Lineární kombinací těchto klasifikátorů získáme silné klasifikátory. [10]

AdaBoost je učící metoda a pro naučení potřebuje velké množství dat pozitivních a negativních identifikací. Následující algoritmus popisuje výběr klíčových slabých klasifikátorů a vytvoření silného klasifikátoru.

1. Pro jednotlivé obrázky $(x_1, y_1)\ \ldots\ (x_n, y_n)$, kde $y_n$ je druh (pozitivní/negativní) obrázku se určí váha podle vztahu
$$w_{1,i} = \frac{1}{2m} \quad a \quad w_{1,i} = \frac{1}{2l}, \tag{11}$$
podle podle druhu obrázku ($m$ je počet negativních a $l$ počet pozitivních).

2. V cyklu se následně provádí

   (a) Normalizují se váhy
   $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}. \tag{12}$$

   (b) Vybere se nejlepší slabý klasifikátor s ohledem na váhu. To je ten, jehož vážená chyba je nejmenší
   $$h_t = arg\ min_{h_j \in H}\ \epsilon_j, \tag{13}$$
   $$\epsilon_j = \sum_{i=1}^{m} D_t(i)\, I\left[y_i \neq h_j(x_i)\right]. \tag{14}$$

   (c) Pokud je vážená chyba větší než 0,5, tak je cyklus ukončen.

   (d) Vypočte se koeficient slabého klasifikátoru.

   (e) Na konci běhu cyklu dojde k aktualizaci váhy $w$. Váha se zvyšuje pro špatně klasifikované měření a snižuje pro správně klasifikované. V dalším průchodu cyklem se budou hledat slabé klasifikátory, které lépe klasifikuje špatná měření
   $$D_{t+1}(i) = \frac{D_t(i)\, exp\left(-w_t y_i h_t(x_i)\right)}{Z_t}, \tag{15}$$
   $$Z_t = \sum_{i=1}^{m} D_t(i)\, exp\left(-w_t y_i h_t(x_i)\right). \tag{16}$$

3. Výsledný silný klasifikátor

$$H(x) = sign\left(\sum_{t=1}^{T} w_t h_t(x)\right). \tag{17}$$

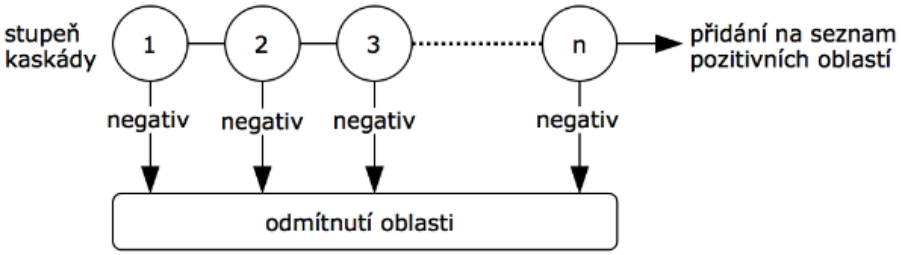
### 3.2.3 Kaskáda klasifikátorů

Při aplikaci klasifikátorů na podokno může dojít k několika možným výsledkům

- True positive (TP) – pravdivě pozitivní - v obrázku je hledaný objekt (tvář),
- False positive (FP) – nepravdivě pozitivní – v obrázku není tvář, ale klasifikátor ji detekoval,
- False negative (FN) – nepravdivě negativní – v obraze je tvář, ale klasifikátor ji nedetekoval.

Po vytvoření silných klasifikátorů metodou AdaBoost přichází jejich využití v podobě vytvoření kaskády klasifikátorů určené k rychlému vyhodnocení podokna v obrázku. Kaskáda je složená ze silných klasifikátorů, které obsahují pouze pár slabých klasifikátorů. Jednotlivé klasifikátory v kaskádě mají za úkol minimalizovat FN identifikace.

Uvažujme pro příklad první klasifikátor v kaskádě. Jeho úkolem není bezchybná detekce obličeje, ale pouze vyřazení obrázku, kde obličej na 100% není. FP detekce takového klasifikátoru může být i 50%, ale FN detekce se blíží 0%. Další klasifikátor už může být trochu přesnější o pár procent v FP detekci, ale FN detekce by měla být stále co nejmenší. Takto navazující klasifikátory tvoří kaskádu, která umožňuje rychle filtrovat podokna obrázků, kde není obličej. Na obrázku Obr. 10 je ilustrován algoritmický postup při vyhodnocení identifikace skrze kaskádu.




Obrázek 10: Kaskáda klasifikátorů [12]

Ideálním výsledkem kaskády klasifikátorů by byl výsledek kdy TP detekce by byla 100% a FP a FN 0%. Pro splnění těchto ideálních podmínek by bylo zapotřebí vytvořit velmi dlouhou kaskádu, která by byla časově neefektivní. Cílem je tedy získat kaskádu klasifikátorů, která je schopna úspěšně detekovat pozitivní detekce a minimalizovat negativní detekce za krátkou dobu. Postupné zpřísňování klasifikátoru má za následek, že dochází k menší pravděpodobnosti FP detekce, přičemž stoupá FN. [10]

#### 3.2.3.1 Tvorba Kaskády

Aplikací metody AdaBoost získáváme klasifikátory jejichž cílem je minimalizovat chybu detekce (FP i FN). Nicméně pro potřeby vytvoření jednotlivých stupňů kaskády je (alespoň zpočátku) žádoucí zvýšit TP i za cenu vyšší FP detekce. Řešením je upravit práh detekce silného klasifikátoru. Vyšší práh produkuje klasifikátory s menší TP i FP. Na druhou stranu nížší práh produkuje klasifikátory s větší TP i FP detekcí. Pro nastavení ideálních hodnot kaskády je zapotřebí najít ideální rovnováhu mezi hodnotou prahu, počtem úrovní a počtem slabých klasifikátorů v jednom silném klasifikátoru.

K tomu existuje celkem jednoduchý algoritmus. Vstupem algoritmu je hodnota maximální hodnota FP detekce $f_i$ a minimální hodnota TP detekce $d_i$. Každá vrstva kaskády je vytvořena algoritmem AdaBoost s určitým počtem slabých klasifikátorů, ten je zvyšován dokud není dosáhnuto potřebných hodnot FP a TP pro danou úroveň kaskády. Toto se provádí dokud celková hodnota TP a FP neodpovídá vstupním hodnotám $f_i$ a $d_i$.

Finální podoba detektoru podle autorů metody je složena z 38 vrstev kaskády obsahující 6060 klasifikátorů.

První klasifikátor v kaskádě obsahuje jen dva příznaky (slabé klasifikátory) a odfiltruje přibližně 50% negativních FP detekcí při zachování skoro 100% obličejů. Druhý klasifikátor obsahuje deset příznaků a odfiltruje přibližně 80% negativních detekcí. V dalších úrovních se zvyšuje počet příznaků a přímo úměrně stoupá i úspěšnost filtrace. [10]

## 3.3 Identifikace obličeje

V této kapitole se práce zabývá identifikací osoby, existujícími metodami a výběrem vhodné metody pro implementaci.

Rozpoznání obličeje je pro člověka velmi jednoduchým úkonem, který mu pomáhá už od nepaměti. Stal se z něj natolik běžný úkol, že si vlastně ani neuvědomujeme, jak často jej děláme. Naproti tomu výzkum automatického rozpoznání obličeje je prováděn teprve od minulého století a stále ještě není u konce.

Pro automatické rozpoznání obličeje existuje několik metod, které se dají rozdělit do tří kategorií [13]

- metody holistického přístupu,
- příznakově orientované metody a
- hybridní metody.

První systémy na automatické rozpoznání byly příznakově orientované, založeny na geometrii tváře. Na fotografie tváří byly umístěny body, které identifikovaly pozice jednotlivých částí obličeje jako jsou uši, oči, nos a ústa. Na základě vzdálenosti a úhlů mezi těmito body byly určeny šablony a následně probíhalo porovnání těchto dat. Hlavní výhodou těchto metod je, že nejsou ovlivněné nepříznivými vlastnostmi obrazu jako například osvětlení. Bohužel slabou stránkou je, že nedosahují dobrého poměru potřebné přesnosti a složitosti výpočtu. Dalším problémem je,

že obličeje, které jsou třeba jen z části zakryté nebo jsou vyfoceny pod jiným úhlem, než čelním velmi zhoršují výsledky detekce. [14]

Později se přešlo na identifikaci obličeje pomocí metod holistického přístupu založených na bázi barev, které prokázaly lepší výsledky. Tyto metody srovnávají různé obličeje srovnáním intenzity jednotlivých pixelů v obraze.

Jednou z těchto metod je metoda Eigenfaces. Ta nahlíží na rozpoznání obličeje z trochu širšího pohledu: Kombinuje existující algoritmy k redukci dimenze a obrazové analýze. Stala se základem pro mnoho dalších metod určených k rozpoznáváni obličeje. Jednou z nich je metoda Fisherfaces, která kombinuje metodu Eigenfaces a Fišerovu lineárně diskriminační analýzu pro zlepšení přesnosti výsledků. Obě tyto metody jsou použity v této práci a jsou popsány v následujících kapitolách. [15]

### 3.3.1 Rozpoznávání pohlaví v obraze

Algoritmů pro určení pohlaví existuje mnoho a jsou založené na rozdílných metodách klasifikace. Mezi nejběžnější patří metoda Eigenfaces, Fisherfaces a metody vyuzivajici LBP, ale existují i další jako např. Fusion of facial strips (FFS) nebo metody využívající Support vector machines (SVM).

Metody založené na PCA dokáží zmenšit objem dat, aniž by docházelo k ztrátě informací a jsou i relativně přesné. Jejich hlavním problémem jsou rušivé elementy jako světlo a změna výrazu tváře, ty dokáží velmi ovlivnit úspěšnost detekce. Tento problém eliminují metody založené na LBP, které dokáží pracovat za horších světelných podmínek.

### 3.3.2 Metody redukce dimenze dat

Jednotlivé metody rozpoznání obličeje využívají pomocné metody pro zmenšení objemu dat pro ulehčení následného zpracování.

**3.3.2.1 Principal Component Analysis (PCA)** Metoda hlavních komponent byla založena a popsána Karlem Pearsonem (1901) a implementována Haroldem Hotellingem (1933). Metoda umožňuje redukovat dimenzi dat a maximalizovat rozdíly vstupních dat, čímž dojde ke zjednodušení při následném zpracování, aniž by došlo ke ztrátě informací. Při rozpoznávání tváří se používá jako lineární transformace využívající vektory vypočtené ze vstupní datové sady.

Prvním krokem je spočítání průměru dat, v mém případě by šlo o průměrný obrázek. Data jsou poté reprezentována jako matice M, kde sloupce představují obrazová data. Poté se získá $Y$ násobením matice $M$ a její transponovanou verzí $M'$. Matice $Y$ představuje vzorovou kovarianční matici, ve které jsou hlavní komponenty vypočteny podle vztahu

$$R^T \left(Y^T\right) R = \Lambda, \tag{18}$$

kde $\Lambda$ je diagonální matice hodnot eigenvalues a $R$ je matice ortonormálních vektorů eigenvectors. Využití této matice je následně popsáno v kapitole 3.3.3.1 o metodě Eigenfaces. [15], [16]

**3.3.2.2 Linear Discriminant Analysis (LDA)** Metoda LDA je velmi podobná metodě PCA s hlavním rozdílem, že k datům přiřazuje třídy. Cílem metody LDA je najít podprostor hodnot, který optimalizuje data podle jejich třídní příslušnosti. Snaží se najít největší rozdíly mezi jednotlivými třídami a jejich podobností.

Výpočetní postup je rovněž podobný metodě PCA. Prvně se vypočítají z vstupních dat x-dimenzionální průměrné vektory mezi rozdílnými třídami. Vypočítají se matice rozptylu mezi všemi třídami. Poté se z matic rozptylu vypočítají eigenvectors a eigenvalues. Snižují se hodnoty eigenvalues a vybírá se k počet eigenvectors s největšími hodnotami eigenvalues za účelem vytvoření $d * k$-dimenzionální matice $W$. Tato matice se použije na vytvoření redukovaného podprostoru $Y$. [15]

### 3.3.3 Metody identifikace obličeje

Po zmenšení objemu dat se může přistoupit k potřebné analýze dat, jejíž cílem je rozpoznání obličeje, konkrétně určení pohlaví osoby z obrazu její tváře.

**3.3.3.1 Eigenfaces** Eigenfaces je metoda holistického přístupu, která analyzuje jasové složky obrazu. Je postavena na nutnosti identifikace komponent v datech, které obsahují nejvíc informací. Z tohoto důvodu se používá Analýza hlavních komponent (PCA). Aplikací této metody získáme z obrazových dat soubor několika normalizovaných obrazů „Eigenfaces“. Ty vyjadřují rozdíly od průměru obrazů z testovací množiny. Představují tedy nějakou odchylku od původního obrazu a dají se využít ke zpětné rekonstrukci. Díky využívání průměru obrazů je zřejmé, že pro co nejpřesnější fungování metody je vhodné mít více obrazových dat jednotlivých osob, ideálně s různými výrazy obličeje pro pokrytí co nejvíce možných výrazů obličeje.

Metoda naráží na problém, který je typický pro obrazové data. Tím je vysoký počet dimenzí. Pro příklad dvourozměrný černo-bílý obrázek o rozlišení 100x100 pixelů odpovídá stejně velké matici, která představuje 10000-dimenzionální vektor. Takto rozměrný vektor je zbytečně velký a s největší pravděpodobností bude obsahovat nadbytečné a redundantní informace. Pro rozpoznání obličeje potřebujeme najít změny a přechody v obraze, tudíž je potřeba se zaměřit na rozdílnost dat.

Metoda tedy redukuje data a snaží se při redukci neztratit žádné důležité informace. Při výpočtu jednotlivých normalizovaných obrazů se postupuje následovně dle algoritmického popisu. [15], [16]

Mějme vektor $X$, pro který platí

$$X = \{x_1, x_2, x_3, \ldots, x_n\}\,;\; x_i \;\in\; R. \tag{19}$$

1. Vypočteme střední hodnotu (průměrný obraz)

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i. \tag{20}$$

2. Vypočteme kovarianční matici $Y$

$$Y = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu) (x_i - \mu)^T. \tag{21}$$

3. Vypočítají se hodnoty eigenvalues $\Lambda_i$ a vektorů eigenvectors viz matice $Y$

$$Y v_i = \Lambda_i v_i, \ i = 1, 2, 3, \ldots, n. \tag{22}$$

4. Vybereme k počet eigenvectors seřazených sestupně podle hodnoty eigenvalue

5. Výslednou matici hlavních komponent získáme podle

$$y = W^T (x - \mu). \tag{23}$$

Původní data se dají znovu rekonstruovat podle následujícího vztahu

$$x = Wy + \mu. \tag{24}$$

**3.3.3.2 Fisherfaces** Oproti algoritmu Eigenfaces pracuje metoda Fisherfaces s využitím třídní specifikace vstupních dat. Je založena na metodě LDA. Princip metody spočívá v maximalizaci rozptylu mezi rozdílnými i podobnými třídami na rozdíl od Eigenfaces, kde metoda PCA maximalizuje celkový rozptyl. K tomu samozřejmě potřebujeme mít ve vstupních datech třídní označení jednotlivých obličejů. Takový přístup umožňuje dosáhnout lepších výsledků v určitých situacích. Vždy záleží na specifikaci vstupních dat, typu klasifikace, kterou chceme provádět a mnoho dalších faktorech.

Algoritmický popis je následující.

Na vstupu mějme vektor $X$ s označením $k$ tříd, popsaný jako

$$X = \{X_1, X_2, X_3, \ldots, X_k\}, \tag{25}$$

$$X = \{x_1, x_2, x_3, \ldots, x_n\}. \tag{26}$$

1. Vypočítá se celková střední hodnota

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i. \tag{27}$$

2. Vypočítá se střední hodnota pro jednotlivé třídy $\mu_i$

$$mu_i = \frac{1}{|X_i|} \sum_{x_j \in X_i} x_j. \tag{28}$$

3. Vypočítají se matice rozptylu prvků rozdílných tříd $S_B$ a matice rozptylu prvků ve stejné třídě $S_w$

$$S_B = \sum_{i=1}^{c} N_i \left(\mu_i - \mu\right) \left(\mu_i - \mu\right)^T, \tag{29}$$

$$S_w = \sum_{i=1}^{c} \sum_{x_j \in X_i} \left(x_j - \mu_i\right) \left(x_j - \mu_i\right)^T. \tag{30}$$

4. Algoritmus poté hledá matici W, která maximalizuje rozdílnost tříd

$$W_{opt} = arg\ max_W \frac{\left|W^T S_B W\right|}{|W^T S_W W|,\ \{w_i | i = 1, 2, \ldots, m\}}, \tag{31}$$

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i,\ i = 1, 2, \ldots, m, \tag{32}$$

Kde $w$ představuje m největších eigenvectors podle eigenvalues $\{\lambda_i | i = 1, 2, \ldots, m\}$ a maximální počet eigenvalues odpovídá počtu klasifikačních tříd $c - 1$.

Tento výpočet skrývá jeden problém, který je často způsoben vstupními daty. Maximální hodnost matice rozptylu prvků ve třídě $S_w$ je $N - c$, kde $N$ je počet prvků ve třídě a $c$ je počet tříd. Problém tkví v situaci, kdy počet prvků $N$ je menší než dimenze vstupních dat (v tomto případě to odpovídá počtu pixelů), takže matice $S_w$ je ve výsledku singulární maticí. Řešením je před 3. krokem aplikace metody PCA na tuto matici a tedy převod matice do $N - c$ dimenzionálního prostoru. [15]

Optimalizační problém v kroku 4 lze následně přepsat jako

$$W = W_{fld}^T W_{pca}^T, \tag{33}$$

$$W_{pca} = arg\ max_W \left|W^T S_T W\right|, \tag{34}$$

$$W_{fld} = arg\ max_W \frac{W^T W_{pca}^T S_B W_{pca} W}{W^T W_{pca}^T S_W W_{pca} W}. \tag{35}$$

### 3.3.4 Local binary patterns histogram

LBPH je metoda rozpoznání tváře patřící mezi metody založené na geometrii tváře. Skládá se z výpočtu LBP a následném zpracování a vytvoření histogramu. LBP je výpočetní metoda

popisující jednotlivé body obrazové textury pomocí binárních hodnot.

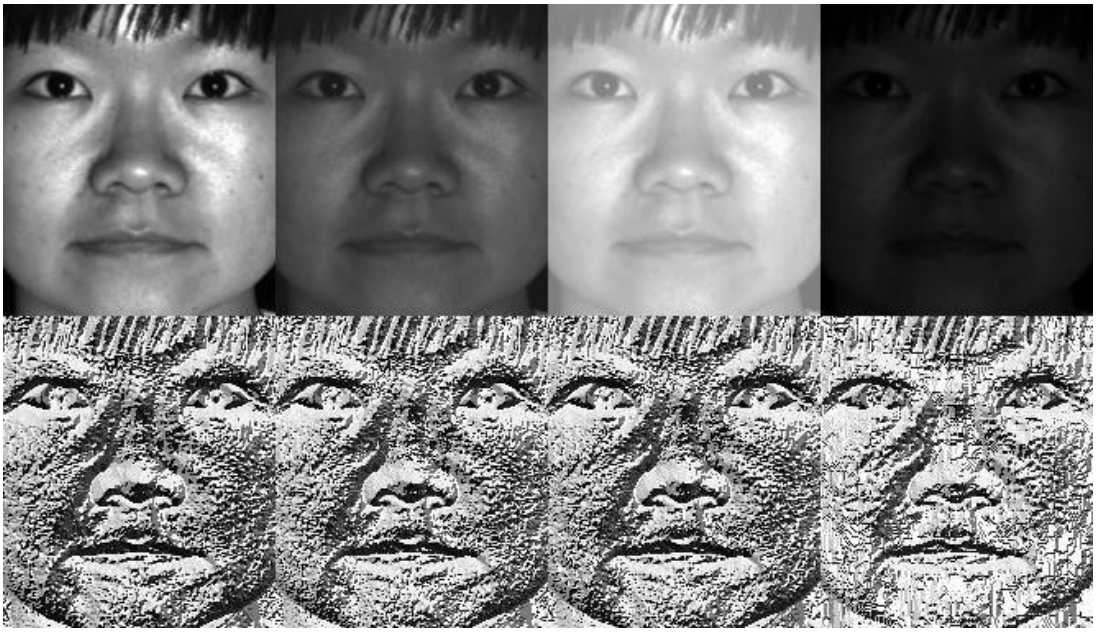
Při výpočtu hodnoty LBP se prochází každý pixel obrázku ve stupních šedi (vyjímaje krajní pixely, protože nemají dostatek sousedních bodů) a přitom se pomocí ohodnocovací funkce získá jeho hodnota. Ohodnocovací funkce pracuje s okolními pixely, které jsou specifikovány pomocí parametru $R$, udávající vzdálenost bodů od aktuálního pixelu, a parametru $P$, který stanovuje počet bodů v daném okolí.

Vzorec pro výpočet hodnoty LPB pro různé varianty prvků $P$ a $R$, kde $g_c$ je centrální pixel a $g_p$ sousední pixely.

$$LPB_{P,R} = \sum_{p=0}^{P-1} s\left(g_p - g_c\right) 2^P, \tag{36}$$

$$s\left(x\right) = \begin{cases} 1 & A, \text{ pro } x \geq 0 \\ 0 & A, \text{ pro } x < 0 \end{cases} \tag{37}$$

Pokud budeme uvažovat výsledné hodnoty opět jako obrázek v 256 stupních šedi, výsledek by mohl vypadat jako na Obr. 11.



Obrázek 11: Výsledek aplikace ohodnocovací funkce LBP algoritmu [19]

Jak je z obrázku patrné, velkou výhodou metody je, že výsledek není razantně ovlivněn, když jsou zhoršené světelné podmínky.

Základní metoda dostala časem určitá vylepšení. Prvním z nich je použití uniformních vzorů. Ta předpokládá, že výsledek LBP je uniformní, pokud obsahuje maximálně dva přechody binárních hodnot z 0 na 1 a zpět. Druhým vylepšením je využití rotační invariance. Díky ní můžeme vybrat nejmenší hodnotu z různých výsledků při natočení původního obrazu (případně natočení

samotného vzoru). Použitím obou vylepšení dokážeme snížit počet vzorů až na $P + 1$, protože zjistíme, že například tyto vzory jsou považovány za stejné: 00000000, 00011110 a 1000001.

Takto reprezentovaný obraz je poté rozdělen na menší části a pro každou tuto část se spočítá histogram podle

$$H_i = \sum_{x,y} I\left\{f_l\left(x,y\right) = i\right\},\ i = 0,\ldots,\ n-1, \tag{38}$$

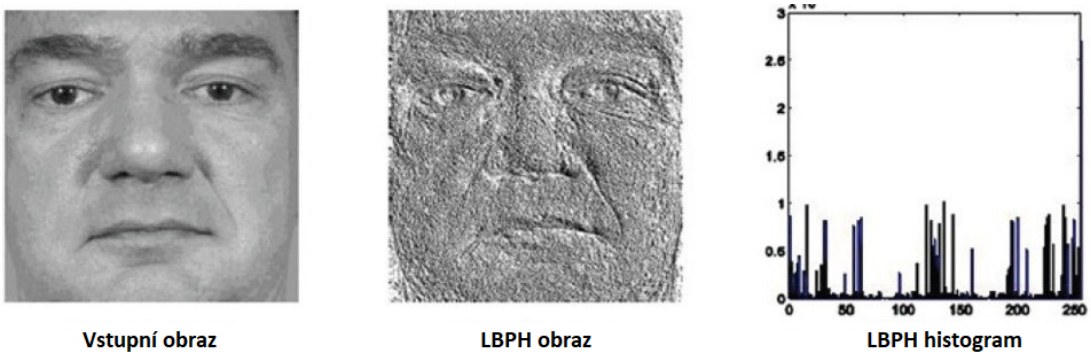
kde $n$ je počet rozdílných výsledných hodnot LBP a funkce $I\left\{A\right\}$ odpovídá

$$I\left\{A\right\} = \begin{cases} 1, & A \text{ je pravda} \\ 0, & A \text{ je nepravda} \end{cases} \tag{39}$$

Tento histogram obsahuje informace o rozložení menších vzorů jako jsou přechody, hrany, body a plochy. Pro efektivní reprezentaci obličeje se musí přidat informace o poloze v rámci obrazu. Obraz je proto rozdělen na části $R_0, R_0, \ldots, R_{m-1}$ a histogram je definován jako

$$H_{i,j} = \sum_{x,y} I\left\{f_l\left(x,y\right) = i\right\} I\left\{(x,y) \in R_j\right\},\ i = 0,\ldots,\ n-1,\ j = 0,\ldots,\ m-1. \tag{40}$$

V tomto histogramu máme uložen popis tváře ve třech úrovních. Hodnoty jednotlivých pixelů, malé oblasti označené podle součtu hodnot pixelů v nich a tyto malé oblasti dohromady tvoří celý obraz tváře. [17]




Obrázek 12: Histogram jasových úrovní obrazu po aplikaci LBPH algoritmu [18]

## 3.4 Obrazová data

Pro potřeby vyzkoušení a otestování popsaných algoritmů bylo použito několik volně přístupných databází fotek obličeje, které značně zjednodušily ověřování funkčnosti a trénování algoritmů pro detekci obličeje. Na stránkách prof. Mislava Grgice a Kresimira Delace z univerzity v Zábřehu v Chorvatsku se nachází přehled databází fotek zaměřených na obličeje. Ačkoli je seznam neaktualizovaný, tak velká část referencí je stále platná. Bylo vybráno několik datasetů, které jsou následně využité při implementaci. [20]

### 3.4.1 AT&T Facedatabase

Tato databáze obsahuje fotky pracovníků laboratoře AT&T v Cambridge pořízené v období mezi dubnem roku 1992 a dubnem roku 1994. Databáze byla využívaná pro projekty rozpoznávání obličeje ve spolupráci s univerzitou v Cambridge.

Obsahuje fotky čtyřiceti různých osob, kde pro každou osobu je deset fotek s různými detaily, jako je nasvícení obličeje, výrazy obličeje. Každá fotka je orientována na obličej s jednobarevným černým pozadím. Obrázky jsou černobílé, uloženy ve formátu PGM s rozlišením 92x112 s 256 odstíny šedi. [21]

### 3.4.2 Caltech Facedatabase

Databáze fotek pravděpodobně pracovníků pořízená Markusem Weberem na univerzitě California Institute of Technology. Obsahuje 450 fotek od dvacetisedmi osob s rozdílnými světelnými podmínkami, výrazy v obličeji apod. Fotky jsou barevné s rozlišením 896 x 592 pixelů. Na rozdíl od předchozích databází neobsahují obrázky pouze obličej.

Proto před samotným využitím pro detekci musely být fotky předzpracovány. Prvním krokem bylo ořezání obrázku. Pro ořezání byla využita metoda detekce obličeje Viola-Jones. Ořezy byly provedeny v poměru stran 1:1 a následně zmenšeny na jednotnou velikost 100 x 100 pixelů. Posledním krokem byl převod do stupňů šedi a vymazání vytvořených obrázků, které detektor vybral a neobsahovaly tvář. [22]

### 3.4.3 BioID Face database

Další využitou databází je databáze z institutu BioID. Obsahuje 1521 černobílých obrázků s obličeji osob obou pohlaví v jednotném rozlišení 384x286 pixelů. Počet obrázků pro jednolivé osoby se liší a osoby rovněž nejsou strukturovány do složek. Strukturalizace, rozdělení dle pohlaví a ořez fotek je podobně jako u Caltech databáze provedeno pomocným programem popsaným v praktické části testování detektoru. [23]

### 3.4.4 Yale Facedatabase

Databáze obsahuje 165 černobílých fotek patnácti osob, přičemž každá osoba má jedenáct různých fotek v různých podmínkách. Obrázky jsou v GIF formátu s rozlišením 320x243. Fotky osob nejsou oříznuty kolem obličeje, takže je stejně jako u předchozích databází provedeno předzpracování a strukturalizace dat. [24]

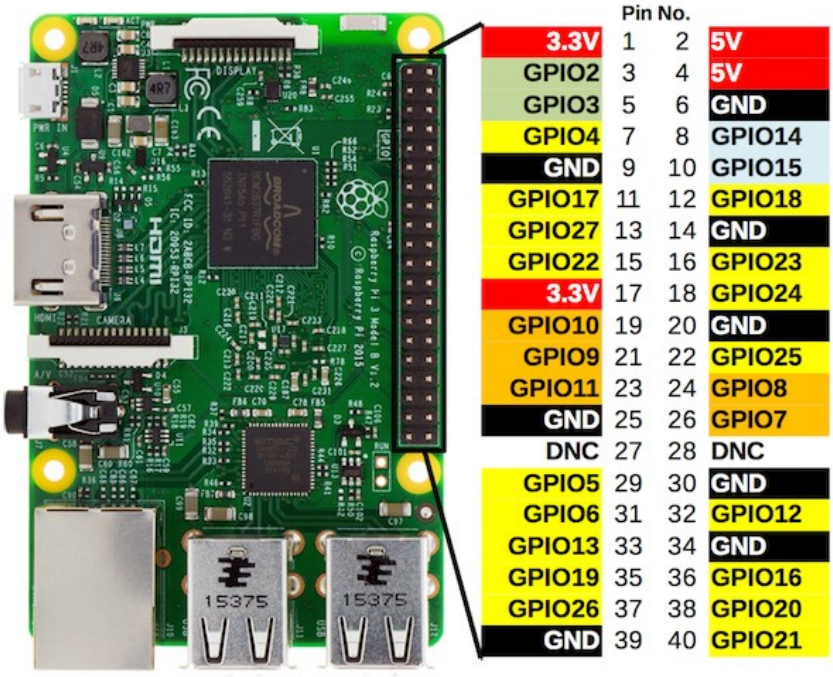
# 4 Použitý hardware a fyzická realizace

První část praktického vypracování popisuje výběr hardwaru umožňujícího realizovat implementaci algoritmů popsaných v teoretické části práce. Některé HW komponenty se v průběhu vývoje projektu měnily, ale popsány jsou jen komponenty využité ve finální podobě robota.

## 4.1 Raspberry Pi

Hlavní komponentou obstarávající funkčnost celého robota musí být relativně výkonný počítač umožňující zpracování audio i video signálů. Dále by měl být malý, aby byl co nejvíce mobilní a zároveň úsporný pro dosažení dlouhé výdrže na baterii. Vybraným počítačem je Raspberry Pi 3 Model B. Raspberry Pi je miniaturní počítač velikosti platební karty, který je založen na System on a Chip (SoC) čipu společnosti Broadcom. Vyvinut byl v roce 2012 britskou nadací Raspberry Pi Foundation s cílem podpořit výuku informatiky na školách. Hlavní předností je jeho velikost a cena, nevýhodou pak výkon, který je oproti klasickým stolním počítačům mnohem menší.

Oproti klasickému PC disponuje možností jednoduše ovládat různá externí zařízení pomocí GPIO rozhraní (většinou různé senzory, čidla atd.). V kombinaci s nízkou energetickou náročností, širokou možností a jednoduchostí použití je to ideální volba pro implementací různých automatizačních projektů. Pro tvorbu projektu byl zvolen model 3 B hlavně z důvodu, že oproti




Obrázek 13: Raspberry Pi 3 Model B a schéma GPIO portu [25]

ostatním modelům má více GPIO pinů a větší výkon. Tento model používá SoC BCM2837, který obsahuje 64-bitový čtyřjádrový procesor ARM Cortex-A53 s frekvencí 1,2 GHz a grafický

procesor Broadcom VideoCore IV. Počítač pracuje s operační pamětí o velikosti 1 GiB. Dále je vybaven čtyřmi USB 2.0 porty pro připojení klasických externích periférií, Ethernet portem a integrovanou Wi-Fi pro připojení k síti a HDMI výstupem pro připojení monitoru/TV. Deska má k dispozici i speciální DSI a CSI konektory pro připojení obrazovky a kamery k Raspberry Pi a již zmíněné GPIO konektory. Hlavní výhodou GPIO konektorů je možnost připojení zařízení přes sběrnici I2C a SPI.

GPIO rozhraní se skládá ze čtyřiceti pinů (Obr. 13), kde většinu tvoří digitální vstupy a výstupy s podporou PWM, dále pak nabízí možnost napájení externích komponent pomocí dvou pinů s napětím 5V a dvou s napětím 3,3V. Některé piny jsou navíc určené pro připojení komponent přes sběrnice I2C a SPI. [26]

## 4.2 Arduino

Arduino je otevřená vývojová platforma, která je založená na jednoduché počítačové desce (hardware) a vývojovém prostředí, které slouží k tvorbě programů (software). Na rozdíl od Raspberry Pi není Arduino určené k nahrazení stolního počítače, ale slouží hlavně k získávání údajů od různých snímačů a senzorů a na základě těchto údajů a vytvořeného programu generuje určitý výstup. Pro vytvoření programu existuje programovací jazyk Arduino, vývojové prostředí Arduino IDE.

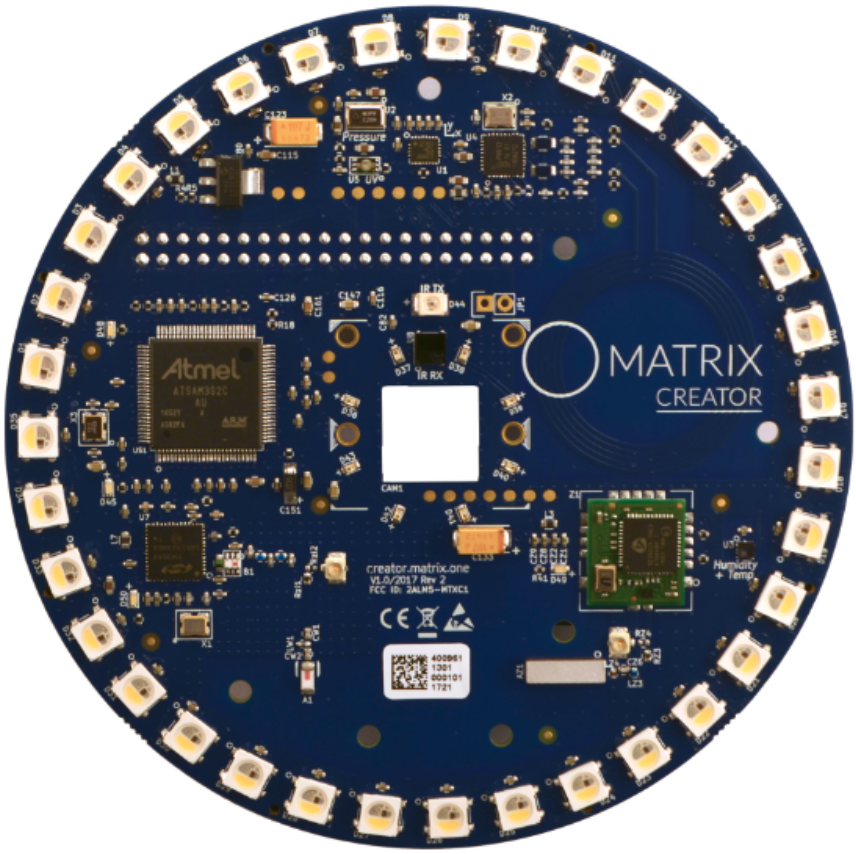
Podobně jako u Raspberry Pi i u Arduina existuje několik různých modelů. Jádrem všech modelů je 8mi bitový mikrokontroler od firmy Atmel. Liší se od sebe množstvím sekundárních obvodů a konektorů. Pro implementaci projektu byl vybrán model Arduino Nano, který vyniká hlavně svou miniaturní velikostí, aniž by přišel o důležité součásti. Má k dispozici stejný počet vstupních a výstupních konektorů jako Arduino Uno a Duemilanove (14 digitálních a 8 analogových).

V tomto projektu plní arduino funkci „prostředníka“. Stará se o čtení digitálních signálů ze senzorů, zpracování těchto signálů a předání výsledků do Raspberry Pi, které na základě těchto údajů provádí hlavní programovou smyčku. Komunikace s Raspberry probíhá sériovou linkou skrze USB port na Raspberry a Mini USB port na Arduinu. Čtení digitálních signálů sice dokáže i Raspberry, které je i mnohem výkonnější, nicméně díky nutné režii operačního systému je v určitých úlohách mnohem pomalejší a nedokáže pracovat přesně v rámci mikrosekund, což je stěžejní pro práci s některými senzory. [27]

## 4.3 Matrix Creator

Matrix Creator (Obr. 14) je HAT (Hardware on Top) vývojová deska pro Raspberry Pi založená na mikrokontroleru ARM Cortex M3, která výrazně rozšiřuje možnosti využití Raspberry Pi jako IoT zařízení. Pro její správné fungování je tedy nezbytné mít Raspberry Pi. Je kompatibilní s modely Raspberry Zero, Raspberry Pi B+ a novější. Deska kompletně zakrývá Raspberry z jeho horní strany a osazuje se na čtyřiceti pinový GPIO port. Naštěstí je na samotném Matrix

Creatoru umístěno dalších 24 pinů, které nahrazují obsazené GPIO piny a v případě potřeby umožňují připojit další zařízení. Vývojáři nabízí možnost pracovat s deskou skrze Matrix Open System, což je open-source systém pro vyvíjení aplikací běžící v node.js. Je to programátorsky více příjemnější verze ovládání desky, ale lehce omezující v rámci využití desky pro specifické úlohy. Naštěstí se dá deska ovládat skrze nízkoúrovňovou vrstvu Matrix Hardware Abstraction Layer, která obsahuje ovladače k jednotlivým komponentám na desce a je psaná v C++.




Obrázek 14: Vývojová deska Matrix Creator [29]

Hlavním využitím desky je zpracování mnoha různých dat z okolího světa, rozšíření možností komunikace s ostatními zařízeními a případná zpětná vazba pomocí LED diod. Na desce je umístěno mnoho senzorů pro čtení fyzikálních veličin: pole mikrofonů složené z osmi MEMS mikrofonů, senzor teploty, 3D akcelerometr, 3D gyroskop, 3D magnetometr, barometr, výškoměr a senzor infračerveného záření. Dále deska rozšiřuje možnosti připojení okolních zařízení k Raspberry Pi tím, že přidává možnost bezdrátového připojení Bluetooth, Zigbee, Z-wave a NFC. Pro interakci a zpětnou vazbu lze využít 35 LED diod umístěných po okraji desky.

V rámci projektu jsou využité LED diody pro signalizaci stavu programu a zobrazení výsledků jednotlivých dílčích úkolů/pokynů. 3D magnetometr se používá jako kompas pouze ve

2D rovině pro orientaci v prostoru. Pole osmi mikrofonů se používá k analýze zvuku, konkrétně k rozeznání slovních povelů a rozpoznání směru odkud přišly. [28] [29]

## 4.4 Vstupně výstupní HW zařízení

V této kapitole je popsán využitý hardware, který neplní řídící funkci programu, ale pouze poskytuje a případně i příjimá data z (resp. do) řídících komponent.

### 4.4.1 Měřič vzdálenosti HY-SRF05

Tento ultrazvukový měřič vzdálenosti umožňuje měřit vzdálenost v rozsahu od 2cm až do 4,5m s přesností v rámci centimetrů. Modul funguje na principu vysílání a zpětné detekce ultrazvukového signálu na frekvenci 40kHz. Měřící úhel je přibližně 45 stupňů, takže měřič je na větší vzdálenosti dost nepřesný. Modul se připojuje dvěma datovými vodiči („trigger“ a „echo“) k Arduinu a dalšími dvěmi k napájení 5V. Napájení je připojeno skrze distrubuované 5V napájení z Arduina, jeden datový vodič slouží ke spuštění měření a na druhém vodiči se měří délka odpovědi, která je přímo úměrná délce vzdálenosti mezi senzorem a předmětem.

Pro spuštění měření je potřeba na pin „trigger“ přivést budící signál o délce minimálně 10us. Následně se provede měření, kdy senzor vyšle osm 40kHz pulzů a v závislosti na rozdílu času mezi vysláním pulzu a vrácením se vypočte vzdálenost. Výpočet vzdálenosti je závislý na rychlosti zvuku, proto se může výsledek lišit podle různých vlastností vzduchu v atmosféře, hlavně tedy vlhkosti, teploty a tlaku. Pro výpočet v mém případě byla použita předpokládaná rychlost zvuku 340 m/s, což zhruba odpovídá rychlosti zvuku při teplotě 15°C v nadmořské výšce 0 m.n.m. Rychlost zvuku v našich klimatických podmínkách se bude lišit jenom v řádu jednotek, takže nijak zásadně neovlivní výslednou vzdálenost. Arduino změří délku signálu na pinu „echo“ a tento naměřený čas následně převede na vzdálenost podle následujícího vztahu

$$d = (t * (340m/s))/2. \tag{41}$$

Výsledná vzdálenost se musí dělit dvěma, protože signál urazil dráhu od senzoru k překážce a zpátky. [30] [31]

### 4.4.2 Infračervený senzor překážek

Modul senzoru překážek funguje na jednoduchém principu vyslání a příjmu infračerveného záření. Modul senzoru překážek má k dispozici čtyři senzory, které se připojují kabeláží k hlavnímu modulu. Každý z jednotlivých čtyřech modulů je osazen infračervenou diodou a fototranzistorem. Citlivost fototranzistoru se dá regulovat čtyřmi samostatnými potenciometry umístěnými na hlavní desce modulu.

Dioda vysílá signál o určité frekvenci a fototranzistor čeká na příjem tohoto signálu. K tomu dojde, když se modul přiblíží k překážce, protože dojde k odražení infračerveného signálu a

hlavní modul vyčte z fotorezistoru informaci o příjmu signálu. Informace ze všech čtyř senzorů je k dispozici na čtyřech výstupních pinech modulu jako logická 0 nebo 1. Arduino čte data z těchto pinů a následně vyhodnocuje přítomnost překážky. [32]

### 4.4.3 Raspberry Pi kamera

Pro Raspberry Pi se dodávají různé moduly jako oficiální příslušenství, jedním z těchto modulů je videokamera, která se připojuje k Raspberry Pi speciálním konektorem CSI. První verze kamery byla vydána v roce 2013 a měla 5 MPx senzor od firmy OmniVision, konkrétně model OV5647. Druhá verze byla vylepšena použitím senzoru IMX219 s rozlišením 8 MPx od Sony. Kromě citlivějšího senzoru má kamera taky lepší kvalitu obrazu, věrohodnější podání barev a lepší obraz při slabém osvětlení. Podporuje rozlišení až 1080p při třiceti snímcích za sekundu. Obě verze kamery se vyrábějí také s podporou nočního vidění, takže při přidání infračerveného osvětlení lze kameru použít i v noci.

Využití kamery v mé práce je určeno výhradně na detekci obličeje osoby v záběru. Pro toto použití je použita novější verze kamery s použitím rozlišení 720p, kvůli rychlosti zpracování obrazu. [26]

### 4.4.4 Mikrofonové pole

Na desce Matrix Creator je umístěno osm MEMS mikrofonů. Jsou umístěny po obvodu desky, připájeny zespod a poslouchající zvuk přicházející shora k desce skrze vyvrtané díry. Jde o všesměrové mikrofony MP34DB02 od firmy STMicroelectronics.

Jejich hlavní výhodou je minimální velikost při zachování rozumné kvality nahrávaného zvuku. Nevýhodou je pak nízká hlasitost nahrávaného zvuku, která limituje použití těchto mikrofonů na větší vzdálenosti. [28]

### 4.4.5 Magnetometr

Magnetometr je součástí desky Matrix Creator, kde je umístěn v čipu LSM9DS1 spolu s akcelerometrem a gyroskopem. Je to součástka umožňující měřit magnetickou indukci, tento konkrétní model ji dokáže měřit ve třech dimenzích.

Základem principu fungování magnetometru je polovodivý plátek, který je ve dvou bodech připojen k elektrickému obvodu a prochází skrze něj elektrický proud. Na elektrony procházející plátkem působí magnetické pole a odchyluje je od nejkratší cesty skrze polovodič. Tato odchylka je měřená a změna této hodnoty představuje změnu v působení magnetického pole.

V projektu se používá k měření magnetického pole Země a je tudíž použit jako 2D kompas. Správné určení světové strany je důležité pro vyhodnocení aktuálního směru, na který je zařízení orientováno. Získaná data z magnetometru, ale můžou v určitých situacích nabývat špatných hodnot. Působí na něj totiž i jiné magnetické prvky, které můžou ovlivnit naměřené hodnoty. Hlavním problémem je přítomnost magnetů v součástkách samotného robota, jedná se především

o elektromotory (případně i reproduktory). Tyto součástky je nutné odstínit, což se řeší vrstvami materiálu, které eliminují vliv těchto součástek na magnetometr, případně zajištěním dostatečné vzdálenosti mezi nimi. [28]

### 4.4.6 Podvozek

Hlavními komponentami podvozku jsou dva elektromotory a H-můstek. Obě tyto komponenty fungují na stejnosměrný proud s pracovním napětím přibližně 5V. Samotné elektromotory jsou jednoduché na ovládání a umí se točit oběma směry. Při přivedení kladného p napětí na první pól a záporného na druhý se motor točí jedním směrem a při přehození pinů se motor točí opačným směrem.

H-můstek je zde použit pro jednodušší ovládání elektromotorů a zároveň problém omezení GPIO výstupů na Raspberry Pi, které nedokáží dodávat dostatek proudu pro provoz elektromotorů. Jeden H-můstek dokáže nezávisle obsluhovat oba motory najednou. Samotný můstek, ale potřebuje k provozu samostatné napájení díky relativně velkému proudovému odběru.

## 4.5 Fyzická realizace

Robot stojí na podvozku tvořeném kulatou deskou z průhledného plastu o průměru 15cm. Stojí na čtyřech kolech z toho dvě jsou statické a dvě jsou otočné sloužící k udržení rovnováhy. Statická kola jsou poháněna motory připevněnými zespod podvozku, které se umí točit oběma směry. Na desce podvozku je umístěn zdroj napájení robota - powerbanka od firmy AData. Powerbanka má kapacitu 10050 mAh a pro připojení má k dispozici dva USB výstupy poskytující 5V napětí pro ostatní moduly. Nad powerbankou je umístěn H-můstek sloužící k ovládání motorů na podvozku. V přední části podvozku jsou navíc umístěny ještě dva infračervené senzory zabraňující možné kolizi při pohybu robota.

Nad spodní deskou tvořící základ podvozku je umístěna stejně velká deska odsazena distančními sloupky pro umístění zbývajících modulů. Na této horní desce je umístěn mikrokontrolér Arduino Nano osazený do nepájivého pole, hlavní deska senzoru pro detekci překážek a měřič vzdálenosti HY-SRF05 v plastovém držáku.

Robot má ještě třetí patro, které je ale bez pomocné plastové desky. Základ tohoto patra tvoří počítač Raspberry Pi, který je uchycen skrze montážní otvory k desce druhého patra distančními sloupky. K počítačí jsou připevněny zbývající moduly: vývojová deska Matrix Creator a kamera pro Raspberry Pi.

### 4.5.1 Seznam komponent

- Základní sestava motorů s prvním a druhým patrem podvozku.
- Čtyři kusy distančních sloupků délky 30 mm o průměru závitu 3 mm.
- Montážní konzola pro uchycení modulu ultrazvukového měřiče vzdálenosti

- Sestava šroubků, distančních sloupků, matek s průměrem 2,5mm.

## 4.6 Distribuce napájení

Pro možnost pohybu robota je nutné zajistit napájení z baterie, v mém případě se jedná o klasickou powerbanku sloužící především k nabíjení chytrých telefonů, tabletů a ostatních mobilních zařízení. Velkou výhodou vybraných komponent je, že všechny potřebují pro svou funkčnost stejné napětí - stejnosměrné o síle 5V. Byla vybrána powerbanka od firmy ADATA, konkrétně model AA10050-5V-CSV. Hlavními kritérii při výběru byla relativně malá velikost, dostačující kapacita, tlačítko pro zapnutí/vypnutí a dva USB výstupy pro zajištění napájení bez nutnosti použití USB rozbočovače. USB výstupy powerbanky jsou podle výrobce v maximálním zatížení schopny dodávat 2,1A na prvním výstupu a 1,0A na druhém, což je dostačující i při maximálním využití robota.

K prvnímu výstupu s větší propustností proudu je připojen H-můstek k ovládání motorů. Na druhém výstupu je připojeno Raspberry Pi ze kterého je 5V napětí distribuováno dále. Napětí přivedené do Raspberry Pi se dále rozvádí skrze USB pro napájení Arduina a z něj se napájí modul se senzory překážek. Z Raspberry se napájí kamera a rovněž deska Matrix Creator, která dále přivádí napětí pro ultrazvukový měřič vzdálenosti.

### 4.6.1 Seznam komponent

- Powerbanka ADATA AA10050-5V-CSV.
- Sada propojovacích kabelů DuPont o různých délkách.
- Tři USB kabely (USB – micro USB, USB – mini USB, USB – 2 vodiče).

# 5 Programovací prostředí

Pro tvorbu a zprovoznění projektu bylo zapotřebí použít několik softwarových řešení, které umožňují implementaci v C++ pro Raspberry Pi a implementaci na platformě Arduino.

## 5.1 Visual Studio 2017 Professional + VisualGDB

Na začátku implementace bylo použito prostředí Codeblocks IDE, které je součástí instalace OS Raspbian pro Raspberry Pi. Pro práci na Raspberry Pi bylo zapotřebí připojit monitor skrze HDMI port a základní periférie přes USB konektory. Toto řešení bylo dostačující, ale nebylo slučitelné s potřebnou mobilitou robota. Dalším krokem byl přístup k Raspberry skrze vzdálenou plochu stále s používáním IDE Codeblocks. Pro realizaci spojení byl vybrán software VNC Server, což je program pracující na architektuře klient-server, který velmi jednoduše umožňuje připojit dva počítače na stejné LAN síti. Výhodou je, že k provozu není potřeba žádná kabeláž, stačí připojit Raspberry ke stejné Wi-Fi síti jako počítač z kterého se k Raspberry Pi přistupuje. Hlavní nevýhodou tohoto připojení je velmi pomalá odezva, díky čemuž nebylo toto řešení využito.

Vybraným řešením pro implementaci bylo použití vývojového prostředí Visual Studio 2017 Professional (dále VS) od firmy Microsoft. VS má pro vývoj dostupných několik verzí, byla použita verze Professional, kde je zapotřebí mít placenou licenci. Tato licence je k dispozici pro studenty na Katedře Informatiky zdarma pro studijní účely. VS umožňuje vyvíjet projekty v mnoha programovacích jazycích pro několik platform. Jako každé jiné IDE i VS zjednodušuje proces vývoje od editace po spuštění programu tím, že některé kroky provádí automaticky a uživatel je od nich odstíněn. Nicméně připojení ke vzdálenému systému a vyvíjení na něm k nim nepatří.

Pro tyto účely byl využit rozšiřující plugin VisualGDB pro VS od firmy SysProgs. Cílem tohoto produktu je právě možnost vzdáleného vývoje projektů skrze VS, kdy VS je použito jako editor na jednom počítači a na druhém probíhá sestavování a ladění. Podporuje vývoj na mnoho platformách, mezi hlavní využití patří vývoj C/C++ aplikací na Linuxu, aplikace pro Android, ESP32 a Arduino. Pro vývoj C++ aplikací na Linuxu je hlavní výhodou využití již existujícího komfortu při vývoji v samotném VS a to především zachování funkčnosti IntelliSense a ladění s využítím vzdálené konzole. Pro svoji funkci vyžaduje připojení mezi oběma počítači, které na počítači s Windows zajišťuje Xming. Xming je open source program poskytující přístup k vzdálenému PC s využitím SSH. V rámci projektu bylo využito pro přenos textových informací při ladění a obrazových dat při ladění s využitím OpenCV.

## 5.2 Arduino IDE

Arduino IDE je programovací prostředí pro vytváření programů pro Arduino. Jedná se o open-source prostředí pro platformy Windows, Linux a Mac OS. Programovací jazyk je Wiring, což

je rozšíření jazyka C/C++ o knihovny, které ulehčují tvorbu programů pro Arduino.

Prostředí je napsané v jazyce Java a je velmi intuitivní. Skládá se z editoru pro psaní zdrojového kódu, hlavní lišty a pár ovládacích tlačítek pro nejčastější úkony. Mezi ně patří verifikace a nasazení kódu, základní operace s projekty a zobrazení monitoru sériové linky.

Pro vývoj na Arduinu je k dispozici rozsáhlá databáze knihoven, které ulehčují práci s různými Arduino shieldy, elektronickými součástkami apod. Knihovny je možné přidávat přes „Správce knihoven", případně ručně nainstalovat knihovnu zabalenou v archivu. Pro většinu knihoven se uvádí i příklad použití, tím se ulehčuje prvotní seznámení s novou knihovnou.

Prostředí je připraveno k okamžitému použití, je zapotřebí pouze nastavit parametry a typ připojení desky. Dále je nutné specifikovat jaký model desky Arduino je připojen a ke kterému portu. V této práci se jedná o desku Arduino Nano s čipem ATMega 328, která je připojena k portu COM4. Číslo COM portu se liší v závislosti na fyzickém USB portu ke kterému je připojeno Arduino k PC a lze jej zjistit ve správci zařízení Windows. Po nastavení připojení byla ověřena funkčnost nahráním „helloworld" programu.

# 6 Softwarová konfigurace robota

V této kapitole je popsána část softwarové konfigurace projektu, která se týká robota samotného. Skládá se z několika kroků, kdy se začíná instalací operačního systému, konfigurací systému a pomocného softwaru.

## 6.1 Operační systém Raspbian

Raspberry Pi představuje plnohodnotný počítač, který nicméně nedosahuje stejných výkonů jako běžné stolní počítače a notebooky, nejen z tohoto důvodu byly pro něj vytvořeny speciální operační systémy. Od Microsoftu vznikl systém Windows 10 IoT Core s podporou Raspberry Pi 2 a 3. Vznikla rovněž odnož Linuxu, konkrétně z platformy Debian vznikl operační systém Raspbian. Ten využívá jako hlavní prostředí PIXEL, což je upravené prostředí LXDE jehož hlavní výhodou je nízká hardwarová náročnost. Zvoleným OS je Raspbian a to hlavně kvůli jednoduchosti a podpoře pro další software, který je v projektu použit.

Samotná instalace je velmi zjednodušená díky pomocnému programu NOOBS, ten provází celou instalací systému, jeho hlavní výhodou je možnost instalovat více systémů na jednu SD kartu. Při instalaci byl NOOBS využit hlavně z důvodu, že již byl předpřipravený na SD kartě přiložené k Raspberry Pi. Instalace systému proběhne automaticky, nakonci je potřeba vytvořit uživatelský účet. [26]

## 6.2 Matrix Ecosystem

Matrix Creator se dodává s několika různými programovacími vrstvami. Záleží na účelu použití a potřebném přístupu k jednotlivým částem vývojové desky. Jak již bylo popsáno Matrix Creator obsahuje spoustu senzorů a pro jejich využití vytvořili vývojáři tři programovací vrstvy.

První z nich, Matrix HAL (Hardware Access Layer), je určena pro přímý přístup k jednotlivým senzorům použitím API v C++ 11. Hlavní výhodou je maximální variabilita při zpracování dat z senzorů pro individuální zpracování v programu. Další vrstvou je Matrix Lite což je abstraktní vrstva nad Matrix HAL, jejíž hlavní výhodou je zjednodušení práce s deskou. Při použití této vrstvy se dá využít programovacích jazyků JavaScript a Python. Poslední vrstvou je Matrix Core (dříve označován jako Matrix MALOS) což je nejuniverzálnější vrstva, protože používá ZeroMQ. ZeroMQ je knihovna poskytující message queue, díky níž se dá s vývojovou deskou pracovat s více než čtyřiceti programovacími jazyky. [28]

Pro ovládání desky byla vybrána nízkoúrovňová vrstva Matrix HAL. Důvodem jsou omezení, které mají ostatní vrstvy, díky nímž například nejde přistupovat k jednotlivým mikrofonům v poli.

### 6.2.1 Instalace a konfigurace Matrix HAL

Instalace se provádí sestavením zdrojového kódu na stránkách Github projektu. Později přibyla i možnost nainstalovat potřebné balíčky z hlavního repozitáře.

- V prvním kroku je potřeba nainstalovat balíček Matrix init a balíčky pro sestavování a spouštění C++ kódu
  ```
  sudo apt-get install cmake g++ git libfftw3-dev wiringpi libgflags-dev
  matrixio-creator-init
  ```
- Následně stačí naklonovat repozitář z Githubu a zkompilovat projekt
  ```
  cd  /
  git clone https://github.com/matrix-io/matrix-creator-hal.git
  cd matrix-creator-hal
  mkdir build
  cd build
  cmake ..
  make -j4 && sudo make install
  ```
- Pro dokončení instalace je nutné restartovat Raspberry Pi
  ```
  sudo reboot
  ```

Pro vyzkoušení funkčnosti vrstvy HAL je k dispozici několik demo aplikací dodaných v balíčku, které testují jednotlivé moduly desky. Většina modulů na desce je funkčních, ale pro využití mikrofonového pole na desce je zapotřebí dodatečná konfigurace popsaná v kapitole 6.3.2. [28]

## 6.3 Moduly pro zpracování zvuku

Pro umožnění zpracování zvuku jsou použité knihovny třetích stran, které pomáhají se zprovozněním mikrofonového pole a analýzou zvuku.

### 6.3.1 Snowboy

Jednou z nich je Snowboy, je to nástroj pro detekci klíčového slova ze zvukového signálu. Byl vyvořen výzkumným týmem KITT.AI, ve kterém se věnují projektům zaměřeným na analýzu a zpracování zvuku. Snowboy se používá jako samostatný modul pro detekci klíčového slova v projektu Alexa Voice Service, který je zaměřen na hlasové ovládání zařízení na různých platformách. Snowboy momentálně běží pouze na unixových systémech a pro použití v projektu nabízí Snowboy svoje API pro jazyk C/C++. Jsou k dispozici API i pro další jazyky jako Java, Python, které jsou generovány skrze SWIG. Knihovny jsou sestaveny pro platformy Debian Jessie, Ubuntu 64bit, Mac OS X 64bit, všechny verze Raspberry Pi, iOS, Android atd.

Projekt není open-source a nikde není dostupný popis implementace a využité technologie. V rámci trénování modelu se používá neurální síť, pro kterou je vstupem zvukový záznam klíčového slova, které má program detekovat a výstupem detekční model. Detektor je pak schopný offline rozpoznávání klíčového slova, ale je natrénovaný na hlasovou frekvenci a způsob vyslovování daného člověka co zvukové podklady pro model nahrál. Samotné trénování není součástí offline knihovny, ale je nutné nahrát klíčová slova skrze průvodce na jejich webových stránkách.

Pro natrénování modelu na oficiálních stránkách stačí zadat název modelu, jazyk ve kterém klíčové slovo řekneme, svůj přibližný věk a pohlaví. Následně je potřeba třikrát nahrát klíčové slovo a model ve formátu .pmdl je vytvořen. Formát pmdl značí „personal model“, tedy model trénovaný na hlas daného člověka. Snowboy podporuje ještě model univerzální (přípona .umdl), který je nezávislý na hlase a měl by mít lepší detekční schopnosti pro vícero uživatelů. Tento univerzální model se pro klíčové slovo vydává jakmile více uživatelů poskytne zvukové data pro stejné slovo. Pro anglické klíčové slova je potřeba 500 nahrávek od uživatelů, pro jiné jazyky je to ještě více. Z tohoto důvodu je v projektu využit pouze osobní model pro všechna klíčová slova. Na konci průvodce si uživatel může otestovat zda Snowboy rozpozná jeho klíčové slovo.

Velkou výhodou použití tohoto detektoru je, že pro detekci nepotřebuje internetové připojení. Tím je zajištěno, že i při neustálém nahrávání zvuku z mikrofonu se data nikam neposílají a je zajištěno soukromí uživatele. Snowboy rovněž podporuje detekování více klíčových slov najednou, což je využito při implementaci. [33]

**6.3.1.1 Instalace** Pro zprovoznění a vyzkoušení je zapotřebí mít připravené Raspberry Pi, k němu mít připojený Matrix Creator pro využití mikrofonového pole a natrénovaný model klíčového slova. V kapitole 6.1 byla popsána příprava Raspberry Pi a v kapitole 6.2 konfigurace nezbytná pro zprovoznění desky Matrix Creator. Jako klíčové slovo pro natrénování a vytvoření modelu bylo vybráno oslovení „R2D2“.

Prvním krokem je instalace nezbytných knihoven. První z nich je PortAudio, což je open-source knihovna pro zpracování vstupních a výstupních audio signálů na různých platformách, včetně Raspbianu. Dalším nezbytným balíčkem je SoX (Sound eXchange). Je to multiplatformní knihovna umožňující převod mezi audio formáty a úpravu zvuku. Vzhledem k tomu, že testování funkčnosti při prvním spuštění je napsané v programovacím jazyce Python, tak je zapotřebí doinstalovat dodatečné balíčky, včetně knihovny PyAudio pro práci se zvukem.

- Instalace nezbytných balíčků
  ```
  sudo apt-get install python-pyaudio python3-pyaudio sox pip
  install pyaudio
  ```
- Funkčnost nahrávání se ověří spuštěním příkazu rec
  ```
  rec temp.wav
  ```

Díky specifickým vlastnostem mikrofonového pole nebylo prvním ověření funkčnosti dokončeno úspěšně. Návod počítá s použitím jednoduchého USB mikrofonu (případně USB webkamery

s mikrofonem), které se velmi odlišují od vlastností mikrofonového pole připojeného přes GPIO port. Pro zprovoznění bylo zapotřebí využít softwaru ALSA, který umožňuje vytvořit virtuální zvukovou kartu, která umožňuje konfiguraci i méně obvyklých zařízení. [33]

### 6.3.2 ALSA

V předchozí kapitole o Snowboy enginu 6.3.1 bylo zmíněno, že pro získání audio signálu se používá software ALSA (Advanced Linux Sound Architecture). Ten má širokou škálu možností jak nastavit parametry získaného zvuku a rovněž umožňuje aplikovat filtry na zvukový vstup. Nevýhodou je, že využití všech těchto možností je omezeno neúplnou dokumentací a chaotickými webovými stránkami projektu. ALSA podporuje mnoho zvukových karet od běžných stereo až po profesionální multikanálové karty.

Před definováním samotné konfigurace mikrofonového pole bylo zapotřebí ověřit, že je k dispozici v seznamu připojených zařízení. K tomuto ověření slouží příkaz *arecord*, který se spouští v terminále.

```
$ arecord -l
**** List of CAPTURE Hardware Devices ****
card 1: Dummy [Dummy], device 0: Dummy PCM [Dummy PCM]
Subdevices: 8/8
Subdevice #0: subdevice #0
...
Subdevice #7: subdevice #7
```

ALSA rozpoznala mikrofonové pole a prezentuje desku jako jednu kartu s osmi dílčími zařízeními. Informace z tohoto výpisu budu následně využita při vytváření konfigurace.

Konfigurace ALSA zařízení se zapisuje do dvou souborů. Jedním je *.asoundrc* umístěný v domovské složce přihlášeného uživatele a druhý soubor obsahuje stejné informace, ale je využíván všemi uživateli */etc/asound.conf*. V konfiguracích se vytváří tzv „pluginy", které představují virtuální zvukovou kartu, která dokáže implicitně měnit parametry datového proudu audio signálu. Na oficiálních stránkách Matrix Labs [28] jsou k vypsány parametry specifikace potřebné pro zprovoznění:

```
Device name: hw:2,0
Rates(Hz): 8000 12000 16000 22050 24000 32000 44100 48000 96000
Channels for each microphone: 1 2 3 4 5 6 7 8
```

Na základě zjištěných informací a s využitím informací na stránkách ALSA projektu [34] byla vytvořena konfigurace využívající všech osmi kanálů mikrofonového pole. Nastavuje mapování Matrix Creator pro nahrávácí plugin *mic*, který je zároveň nastaven jako výchozí zařízení, takže při nahrávání nemusíme specifikovat ze které karty chceme nahrávat. Tento plugin obsahuje

Tabulka 2: Použité moduly z knihovny OpenCV

| Modul | Funkce |
|---|---|
| Core | Definuje všechny základní moduly a funkce |
| Highgui | Jednoduché GUI určené nejen pro zobrazení obrazu |
| ImgProc | Zpracování/úprava obrazu (filtrace, transformace, převody barev apod.) |
| ObjDetect | Klasifikátory pro detekci objektu v obraze |
| Face | Poskytuje pomocné funkce při analýze obličeje v obraze |

tzv. „slave" plugin *array*, který specifikuje HW označení desky. Soubor s konfigurací je součástí přílohy práce.

Funkčnost konfigurace je následně ověřena nahráním zvuku z mikrofonového pole spuštěním příkazu *arecord* v terminále.

```
arecord test.wav -f S16_LE -r 16000 -d 5
```

Příkaz vytvoří zvukový soubor ve formátu wav o délce pěti sekund, který používá formát ukládání S16_LE (16bit, Little endian) a vzorkovací frekvenci 16000Hz. Následně byla pomocí příkazu *aplay* ověřena funkčnost. [34]

## 6.4 Modul pro analýzu obrazu

### 6.4.1 OpenCV

Pro pomoc se zpracováním obrazu je využita část knihovny OpenCV ve verzi 3.4.0. OpenCV je open-source knihovna zaměřena na počítačové vidění a práci s obrazem s využitím strojového učení. Je vydaná pod licencí BSD, takže je volně přístupná pro studijní i komerční účely. Je napsaná v jazyce C++ a podporována na mnoha platformách, poskytuje rozhraní pro většinu aktuálních programovacích jazyků jako Java, Python, C++ atd.

Knihovna je složena z několika modulů lišících se zaměřením na jednotlivé kategorie analýzy a zpracování obrazu. Jednotlivé moduly jsou přehledně zdokumentované a dostupné na oficiálních stránkách OpenCV. Rovněž jsou zde i příklady pro implementaci reálných úkolů s využitím modulů knihovny, jejichž části posloužily při tvorbě vlastní implementace. [35]

V práci jsou využité jak základní OpenCV moduly, tak i speciální moduly obsahující klasifikátory a analytické metody zaměřené na detekci obličeje. Každý modul má v API vlastní hlavičkový soubor.

**6.4.1.1 Instalace** Instalace je jednoduchá, zahrnuje naklonování repozitáře z GitHubu a sestavení na Raspberry Pi. Po dokončení instalace byla ověřena funkčnost využitím základních modulů v implementaci. Postup pro instalaci je následující:

- Nejprve je zapotřebí mít připraven kompilátor GCC, nástroj pro sestavování CMake, Git a několik dalších balíčků

```
sudo apt-get install cmake git libgtk2.0-dev pkg-config
libavcodec-dev libavformat-dev libswscale-dev
```

- Následně stačí naklonovat repozitář z Githubu a zkompilovat projekt

```
git clone https://github.com/opencv/opencv.git
cd  /opencv
mkdir build
cd build
cmake -D CMAKE_BUILD_TYPE=Release -D
CMAKE_INSTALL_PREFIX=/usr/local ..
make -j4
sudo make install
```

# 7 Implementační část na Arduino Nano

První část softwarové implementace se bude zabývat zpracováním dat na Arduino Nano. Cílem programu vytvořeného pro Arduino Nano je získání dat z ultrazvukového měřiče vzdálenosti a z infračerveného senzoru překážek. Tato získaná data se musí dostat do Raspberry Pi přes USB kabel skrz který probíhá jednosměrná sériová komunikace.

Infračervené senzory detekují překážku ve vzdálenosti přibližně pět centimetrů před robotem. Pokud se objeví překážka, tak na výstupu modulu lze číst hodnotu logické 0, při absenci překážky je zde logická 1. Čtení dat z ultrazvukového měřiče vzdálenosti je rovněž velmi jednoduché. Prvním krokem je spuštění měření vysláním obdélníkového impulzu o délce minimálně 10 $\mu$s na pin TRIG. Ihned poté lze na pinu ECHO detekovat puls, jehož délka je přímo úměrná vzdálenosti mezi senzorem a překážkou. Popis výpočtu je uveden v kapitole 4.4.1.

Informace z obou senzorů jsou posílány sériovou komunikací do Raspberry Pi. Parametry sériové komunikace musí být na obou zařízeních stejné. Pro přenos je použita znaková rychlost 57600 Bd. Pro úspěšné předání informace je rovněž definováno pořadí a počet bitů. Používám běžný druh komunikace 8N1, takže zpráva obsahuje osm datových bitů (8), žádný paritní bit (N) a jeden stop bit (1). Zpráva poslaná sériovou linkou obsahuje vzdálenost naměřenou ultrazvukovým senzorem a součet výstupů z infračervených senzorů oddělené středníkem. Pokud se nepodařilo načíst vzdálenost a zároveň není před senzory žádná překážka, tak se žádné data neposílají, protože by pro ně nebylo využití.

# 8 Implementační část na Raspberry Pi

Hlavní částí softwarové implementace je vytvoření programu v C++ implementující popsané algoritmy v teoreické části. Zdrojový kód je psaný v jazyce C++ ve verzi 11 s využitím prvků z jazyka C z důvodu přizpůsobení některým API modulů třetích stran. Při vývoji byl kladen důraz na vývoj podle *Google C++ Style Guide*, což je sbírka progkterý určuje vývojářské konvence pro zjednodušení čtení a pochopení kódu. Zdrojový kód je okomentovaný a je součástí přílohy této práce včetně ostatních souborů souvisejících s vývojem.

## 8.1 Struktura projektu

Projekt obsahuje jeden hlavičkový soubor, dva soubory se zdrojovým kódem, jeden konfigurační CMake soubor, několik knihoven a hlavičkových souborů knihoven třetích stran a soubory obsahující data nezbytné pro analýzu. Hlavními soubory, jejichž obsah bude popsán v této kapitole jsou

- *main.cpp*
- *utils.cpp* a *utils.h*
- *CMakeLists.txt*

### 8.1.1 Konfigurace pro sestavení

Pro zpřehlednění a zjednodušení práce při sestavování se nepoužívá klasického zápisu v terminále pomocí spuštění kompilátoru gcc a vypsání všech potřebných parametrů. Místo toho je použitý nástroj CMake, který umožňuje sepsat sadu příkazů nutnou pro sestavení do jednoho konfiguračního souboru a pomocí něj pak sestavovat projekt. Konfigurační soubor má název *CMakeLists.txt* a je umístěn ve složce spolu se zdrojový kódy. Část konfiguračního souboru vygeneroval nástroj VisualGDB, nalezení a zahrnutí přídavných knihoven a s tím související proměnné byly doplněny do konfigurace ručně. Obsahuje název projektu, použitou verzi C++, pokyny pro kompilátor, seznam použitých knihoven a zdrojových souborů. Ukázka ze souboru:

```
set (CMAKE_CXX_STANDARD 11)
cmake_minimum_required(VERSION 2.7)
project(rpi_r2)
set(SNOWBOY_LIB "/home/pi/snowboy/include")
...
add_executable(rpi_r2 rpi_r2.cpp utils.cpp utils.h RS-232/rs232.c
RS-232/rs232.h)
...
target_link_libraries(rpi_r2 $FFTW_LIBRARIES)
target_link_libraries(rpi_r2 $OpenCV_LIBS )
```

## 8.2 Vstupy programu

V kapitole 6 byla popsána konfigurace robota nezbytná pro jeho zprovoznění. V této kapitole bude popsána příprava vstupních dat pro trénování algoritmů.

### 8.2.1 Analýza obrazu

Jako vstup pro trénování klasifikátorů na analýzu obrazu jsou využity databáze obličejů popsány v kapitole 3.4. Jde většinou o relativně malé černobílé obrázky obličejů osob snímaných kamerou za různých podmínek osvětlení. Pro jednotlivé databáze se mění velikost obrázků a v závislosti na velikosti vstupních dat se mění i velikost natrénovaného modelu. Nevýhodou trénování je, že je časově náročné. Na druhou stranu jej stačí spustit jednou a výstupy trénovacího algoritmu lze uložit na disk. Pro následné využití klasifikace stačí načíst model z disku a použít ho při vyhodnocování.

**8.2.1.1 Příprava vstupních dat** Vytvoření modelu zahrnuje přípravu vstupních obrázků. Následující postup přípravy se liší podle použité databáze, ale princip zůstává stejný. Prvním krokem je načtení fotek pro natrénování klasifikátoru. Vzhledem k tomu, že obrázků je v jednotlivých databázích hodně, tak ruční načítání v kódu by bylo časově náročné. Pro zjednodušení byl vytvořen program, který načte z určené složky a jejích podsložek obrázky a podle toho, ve které podsložce se nachází mu přiřadí číselnou hodnotu podle pohlaví. Výstupem je CSV soubor obsahující cestu k obrázku na disku a číselnou hodnotu určující kategorii pro klasifikátor oddělené středníkem. Ukázka výstupu:

```
/home/pi/faces/female/s10/1.pgm;100
/home/pi/faces/female/s10/2.pgm;100
...
/home/pi/faces/male/s11/1.pgm;3
/home/pi/faces/male/s11/2.pgm;3
```

Tento projekt *generate_csv_gender* je psán v jazyce C# a byl spouštěn na vývojovém PC s Windows. Zdrojový kód je součástí přílohy práce.

**8.2.1.2 Trénování modelu** Po vytvoření souboru následovalo natrénování modelu. Pro tento úkol byla využita knihovna OpenCV, která obsahuje metody pro trénování všech klasifikátorů popsaných v teoretické části.

Prvním krokem k trénování modelu je načtení trénovacích dat z CSV souboru popsaném v předchozí kapitole. K tomuto slouží metoda *ReadCsv*, která má jako vstupní parametr cestu k CSV souboru. Metoda prochází jednotlivé řádky v CSV souboru a načítá obrázky a názvy tříd do polí. Tyto pole jsou výstupem metody, kdy jsou předávány referencí jako vstupní argumenty funkce.

```
void ReadCsv(string face_list_file, vector<Mat>& images, vector<int>& labels);
void FaceModelTraining(string face_list_file, string model_file, model_type
    type, Ptr<cv::face::FaceRecognizer> model);
void FaceModelTesting(string face_list_file, string model_file, model_type type
    , Ptr<cv::face::FaceRecognizer> model, Size resizing_dims);
void FaceCrop(string haar_file_path, string base_folder, string input_file);
```

Načtení fotek metodou *ReadCsv* je první krok v definici metody *FaceModelTraining*. Ta slouží k natrénování klasifikátoru umožňujícího rozpoznání pohlaví osoby. Typ klasifikátoru (Eigenfaces, Fisherfaces, LBPH) je určeno v inicializační metodě, která je popsána později v implementační části. Operace trénování modelu je výpočetně a tudíž i časově náročná. Výstupy trénování se dají uložit jako XML soubor obsahující klasifikátory pro jednotlivé třídy, takže není nutné provádět trénování při každém spuštění robota. Metoda tedy na konci uloží výstupy trénování do souboru jehož název je specifikován v parametru *model_file* jako vstupní parametr metody.

Poslední pomocnou metodou je metoda *FaceCrop*, která využívá metody detekce Haarových příznaků k nalezení obličeje v obraze a jeho následnému ořezání. Tyto ořezané obličeje jsou uloženy jako nové obrázky v podsložce *crop*. Metoda slouží pouze k ořezu fotek a tudíž byla volána jen párkrát a to při potřebě zpracování obrazových datasetů, které neobsahovaly oříznuté fotografie. V metodě jsou nastavitelné parametry ovlivňující názvy výsledných fotek, jejich rozlišení a barevné spektrum.

### 8.2.2 Analýza zvuku

Příprava dat u analýzy zvuku zahrnuje nahrání a natrénování klíčových slov pro Snowboy knihovnu. Narozdíl od OpenCV se u Snowboy knihovny používají webové stránky na které se nahrává zvuková nahrávka obsahující klíčové slovo. Nelze tedy provést natrénování nového slova bez připojení k internetu.

Nahrávání probíhá na oficiálních stránkách Snowboy projektu [33], kde po přihlášení uživatel získá přístup k seznamu již nahraných klíčových slov. Pro každé klíčové slovo lze nahrát vlastní nahrávku a tím přispět pro zlepšení trénovaného modulu. Postup pro přidání nového klíčového slova je již popsán v kapitole 6.3.1. Tímto postupem byly vytvořeny následující příkazy: „R2D2“, „forward“, „backward“, „move left“, „move right“, „camera“, „stop“. Pro každý z nich jsou k dispozici tři nahrávky ve formátu WAV a jeden personální model ve formátu PMDL v přílohách práce.

## 8.3 API importovaných modulů

V této kapitole budou popsány rozhraní modulů třetích stran využité v projektu pro jednodušší pochopení samotné implementace.

### 8.3.1 Matrix HAL

Jak již bylo zmíněno Matrix HAL je API pro přístup k hardware desky Matrix Creator. Hlavičky ani kód bohužel nejsou okomentované a k HAL vrstvě neexistuje kompletní dokumentace, takže pro zjištění funkcionality metod je potřeba projít a pochopit kód. V repozitáři na GitHubu je k dispozici několik demo aplikací, které ulehčují pochopení funkčnosti některých modulů a metod. Pro ovládání jakéhokoliv modulu na desce slouží samostatný objekt. Každý z nich má k dispozici metodu *Init*, která je potřebná pro inicializaci modulu na desce. Metoda vrací informaci, zda je modul připraven k použití. V projektu nejsou využity všechny dostupné senzory a moduly, mezi použité moduly patří

- *MatrixIOBus* – objekt obstarávající vzájemnou komunikaci všech modulů na desce
- *Everloop* – poskytuje metody pro ovládání LED diod po obvodu desky. Pro změnu osvětlení potřebuje vstupní objekt *EverloopImage*
  - *EverloopImage* - Objekt obsahující pole třicetipěti objektů *LedValue*. Ty obsahují čtyři různé proměnné pro ovládání RGBW diod. Tedy barvy červená, zelená, modrá a bílá. Tyto barvy můžou svítit i zároveň a každá jednotlivá barva má intenzitu svícení v rozsahu od 0 do 255.
- *IMUSensor* – objekt umožňuje čtení naměřených hodnot z 3D magnetometru, 3D akcelerometru a 3D gyroskopu.
- *MicrophoneArray* – umožňuje přístup ke zvukovým datům nahraným mikrofony na desce. Umožňuje nastavit parametry získávaného zvukového signálu mezi které patří např. vzorkovací frekvence a zesílení.

### 8.3.2 Snowboy

Projekt Snowboy není open-source a tudíž nejde stejně jako u Matrix HAL projít zdrojový kód a analyzovat funkcionalitu. Tutoriál na oficiálních stránkách představuje demo v jazyku Python a poskytuje základní přehled o fungování knihovny. V repozitáři na GitHubu jsou k dispozici demo aplikace pro různé jazyky, včetně C++ a v hlavičkovém souboru *snowboy-include.h*, který prezentuje API knihovny, jsou okomentované metody. Je použita aktuálně poslední verze 1.3.0, která byla vydaná 19.2.2018.

Pro spuštění a ovládání knihovny je k dispozici jedna třída typu *SnowboyDetect*, která obsahuje všechny nezbytné metody. Mezi hlavní z nich, které jsou využité při implementaci patří

- *SnowboyDetect* – konstruktor objektu jehož vstupním parametrem jsou názvy modelů klíčových slov,
- *SetSensitivity* – nastavení citlivosti detekce pro jednotlivá slova,

- *RunDetection* – provádí detekci klíčových slov. Vrací celé číslo určující zda došlo k detekci a případně jaké slovo bylo detekováno,
- *SetAudioGain* – umožňuje zesílit vstupní signál z mikrofonu.

### 8.3.3 OpenCV

OpenCV je knihovna umožňující analýzu obrazu, konkrétně detekci a rozpoznání pohlaví osoby z obličeje. Pro tyto úkoly jsou k dispozici různé klasifikační třídy, které pomáhají s implementací analytických metod uvedených v teoretické části práce. Je využita verze knihovny 3.4.0, pro kterou je dostupná kompletní dokumentace a rovněž několik tutoriálů jak s knihovnou pracovat. Použité moduly popsané v kapitole 6.4.1 obsahují spousty tříd, z nichž byla využita pouze malá část. Mezi hlavní patří

- *Mat* – objekt reprezentující matici, je použit k uchovávání obrázků,
- *CascadeClassifier* – kaskádový klasifikátor určený k detekci obličeje v obraze,
- *FaceRecognizer* – rodičovská třída, jejíž potomci obsahují klasifikátory metod určených k rozpoznání obličeje.

## 8.4 Popis implementace

Projekt se skládá z několika metod jejichž předpisy jsou uloženy v hlavičkovém souboru *utils.h* a implementace v souboru *utils.cpp*. V hlavičkovém souboru jsou umístěny direktivy preprocesoru pro přidávání dalších hlavičkových souborů a definice maker, globální proměnné a deklarace metod.

Hlavičkový soubor *utils.h* se vkládá do obou cpp souborů se zdrojovým kódem. Pro zabránění dvojitého vložení se používá ochrana pomocí definice makra a kontroly na jeho definici.

Součástí hlavičky je definice vlastních struktur. První z nich je *CompassData*, která obsahuje kalibrační data z modulu kompasu, které se používají při výpočtu směru. Druhou je pomocná struktura *HotwordInfo*, která obsahuje číslo detekovaného klíčového slova a index mikrofonu, ke kterému dorazilo slovo jako první.

```
struct CompassData{
   float mag_max_x;
   float mag_max_y;
   float mag_min_x;
   float mag_min_y;
   float mag_off_x;
   float mag_off_y;
};
```

```
struct HotwordInfo{
   int micIndex;
   int hotwordId;
};
```

Objekty jednotlivých knihoven jsou vytvořeny jako globální proměnné, což je hlavně z důvodu zjednodušení a odstranění nutnosti předávat tyto objekty mezi metodami.

```
extern hal::MatrixIOBus bus;
extern hal::GPIOControl gpio;
extern hal::Everloop everloop;
extern hal::IMUSensor imu_sensor;
extern hal::MicrophoneArray mics;
extern int led_offset[];
extern bool moving;
extern int min_distance;

extern cv::Ptr<cv::face::FaceRecognizer> model;
extern cv::CascadeClassifier haar_cascade;
extern cv::Size im_size;
```

Zbytek hlavičkového souboru tvoří deklarace metod a na konci souboru je uzavření ochrany proti dvojitému vložení.

### 8.4.1 Inicializace objektů

Skoro každý modul, senzor a komponenta se musí před použitím nastavit. V předchozí kapitole o API jednotlivých modulů jsou popsány inicializační metody, které jsou vždy spuštěny jako první po spuštění programu.

Metoda Initialize provádí inicializaci všech modulů a proměnných (kromě globálních). Je spuštěna jen jednou a to na začátku hlavní *main* funkce programu. Metoda *ChassisIntro* slouží ke kalibraci kompasu. Vstupním parametrem je prázdný objekt *CompassData*. Metoda po spuštění začne otáčet podvozkem a objekt *CompassData* naplní naměřenými maximálními a minimálními hodnotami síly magnetického pole. Zavolání této metody je nezbytné pro správnou funkci kompasu a je rovněž prováděno pouze jednou.

```
void ChassisIntro(CompassData &comp);
void Initialize(string haar_cascade_file, string faces_file, Size image_size,
    snowboy::SnowboyDetect& detector);
```

Poslední dvě inicializační metody *InitializeRS232* a *CloseRS232* slouží k zavedení a ukončení sériové komunikace mezi Raspberry Pi a Arduinem Nano. Sériová komunikace probíhá pomocí knihovny RS-232 od Teunis van Beelena. Knihovna emuluje sériovou komunikaci na USB

Tabulka 3: Kombinace impulsů pro pohyb motoru

| Hodnota pin 1 | Hodnota pin 3 | Pohyb |
|---|---|---|
| 0 | 0 | žádný |
| 0 | 1 | dozadu |
| 1 | 0 | dopředu |
| 1 | 1 | žádný |

portu Raspberry Pi. Vstupem inicializační metody je nastavení způsobu komunikace. Parametry komunikace jsou definovány stejné jako v projektu spouštěném na desce Arduino popsané v kapitole 7. [36]

```
int InitializeRS232(int port_nr);
void CloseRS232(int port_nr);
```

### 8.4.2 Ovládání podvozku

Pro ovládání podvozku byly vytvořeny dvě metody. První z nich *MoveChassis* má definován jeden celočíselný vstupní parametr určující jakým směrem se má podvozek pohnout. Povolené směry jsou definované jako makra v hlavičkovém souboru *utils.h*. Pohyb jednotlivých motorů se určuje nastavením dvou hodnot na výstupní GPIO piny, které ovládají H-můstek. Pravý motor se ovládá přes piny 1 a 3, levý přes piny 5 a 7. Pohyb pravého motoru se řídí podle tabulky Tab. 3. Levý motor se ovládá stejně, pouze díky umístění v opačné poloze jsou směry invertovány.

```
void MoveChassis(unsigned char direction);
```

Další definovanou metodou je *TurnToFace*, která má na vstupu kalibrované hodnoty kompasu a index mikrofonu, ke kterému dorazil hlas uživatele jako první. S využitím těchto hodnot se rozhodne směr otočení (vlevo, vpravo) a dojde k otočení robota k uživateli (index mikrofonu). Vzhledem k občasné nepřesnosti naměřených hodnot kompasu je implementována i pojistka ve formě počítadla, která zajistí, aby nedošlo k zacyklení.

```
void TurnToFace(CompassData &comp, int face_index);
```

### 8.4.3 Využití LED diod

Pro ovládání LED diod je zapotřebí využít třídu *Everloop* z API knihovny Matrix HAL. Po vložení nezbytného hlavičkového souboru a provedení inicializace objektu lze třídu využít pro změnu hodnot svítivosti jednotlivých barevných složek LED diod umístěných po obvodu desky Matrix Creator. Hodnoty svítivosti diod se ukládají do objektu *EverloopImage*, ten je vstupním parametrem pro zápis změn hodnot LED diod. Popis objektu *EverloopImage* a jeho obsahu je v kapitole 8.3.

Pro ovládání osvětlení LED diod bylo vytvořeno několik metod, které jsou určeny většinou k prezentaci výsledků analytických metod popsaných v následujících kapitolách. Vzhledem k tomu, že všechny provádí podobnou činnost, budou uvedeny jen některé z nich. Metoda *LedsByMicIndex* rozsvítí LED diody podle indexu mikrofonu. Index diody, která je nejblíž danému mikrofonu je definován v poli *led_offset* v *main.cpp* souboru. Metoda ve výsledku rozsvítí 7 zelených diod podle definovaných intenzit v poli *lut* a po sekundě je zhasne.

```
void LedsByMicIndex(int mic_index);
```

Druhou metodou pro popis použití objektů je metoda *LedsCircle*. Vstupní parametry metody je procentuální podíl určující kolik diod se rozsvítí v intervalu $\langle 0; 1 \rangle$, barva diod a intenzita svitu. Používá se pro uvědomění uživatele o odpočtu času.

```
void LedsCircle(double percentage, char color, int pwr);
```

### 8.4.4 Obrazové metody

Jedinou metodou pro zpracování obrazu je *GenderFaceCount*. Vstupem metody je časový údaj určující jak dlouho má metoda provádět analýzu obrazu z kamery. Cílem metody je rozpoznat obličeje a v závislosti na počtu osob a jejich pohlaví rozsvítit červené a modré LED diody.

```
void GenderFaceCount(int duration);
```

Prvním krokem běhu je inicializace kamery. Pokud inicializace selže, tak se vypíše chybová hláška do výstupu konzole. V opačném případě se načtou data z kamery do matice reprezentované datovým typem *Mat* z knihovny OpenCV. Následně se provede převod do černobílé a detekují se obličeje v obraze s využitím Haarových příznaků. Jednotlivé nalezené obličeje se ořízno na stejnou velikost jako měly vstupní trénovací data a následně se provede klasifikace vybraným klasifikátorem. Pokud je detekován muž, tak se do pole LED diod přidá jedna modrá LED dioda, v případě detekce ženy červená. Toto pole LED diod je následně zapsáno přes *Everloop* objekt a počet modrých a červených diod reprezentuje počet osob obou pohlaví.

### 8.4.5 Zvukové metody

Pro analýzu zvuku byla vytvořena jedna metoda, jejíž cílem je rozpoznávat klíčové slovo, které uživatel řekl a určit směr odkud zvukový signál k desce dorazil. Vstupem metody *ListenToHotword* je objekt Snowboy enginu a informace o předešlém detekovaném slově. Výstupem je pak informace o detekovaném slově a index mikrofonu, ze kterého přišel zvuk.

```
HotwordInfo ListenToHotword(snowboy::SnowboyDetect& detector, int last_word);
```

Velkou část metody tvoří cyklus, ve kterém se provádí následující postup:

1. V prvním kroku se načtou zvuková data z mikrofonového pole.

2. Následuje výpočet určující směr zvukového signálu

   (a) Tato informace se mění velmi rychle podle velikosti pole s daty a vzorkovací frekvence. Pro pole o 512ti prvcích a vzorkovací frekvenci 16000Hz se hodnota změní každých 0,032 sekundy.

   (b) Z tohoto důvodu se ukládají výsledky detekce směru do jednoduchého cyklického pole obsahujícího posledních padesát výsledků určení směru.

   (c) Následně se provede korelace transformovanych signalu $A'$ a $B'$. Jednotlivé hodnoty signálu $A'$ se vynásobí komplexně sdruženými hodnotami signálu $B'$ a vznikne výsledný signál $C'$.

   (d) V dalším kroku je aplikována inverzní Fourierova transformace na signál $C'$ pro získání výsledku korelace.

   (e) Ve výsledném signalu $C$ jsou hodnoty korelace signalu $A$ a $B$. Amplituda funkční hodnoty signalu $C$ představuje okamžik maximální podobnosti signálu $A$ a $B$. Hodnota ve které je amplituda funkce označuje hodnotu posunu mezi signály $A$ a $B$.

3. V posledním kroku se provede detekce klíčového slova.

   (a) Pokud došlo k detekci slova "R2D2", tak se vypočte medián z cyklického pole v bodě 2b. Výsledná hodnota označuje index mikrofonu určujícího směru hlasu a spolu s hodnotou klíčového slova jsou tyto dvě informace metodou navráceny ve struktuře *HotwordInfo*.

Kromě těchto kroků fukce provádí i vizualizaci čekání, odpočtu a signalizaci směru, přičemž využívá metody pro ovládání LED diod.

### 8.4.6 Metody ostatních modulů a pomocné metody

Pro zpracování dat z ultrazvukového modulu HY-SRF05 a z infračervených modulů detekce překážek je již vytvořen program pro Arduino, který zpracovává signály, provádí převod jednotek a posílá data jako řetězec znaků do Raspberry Pi skrze sériovou komunikaci. Pro zpracování tohoto řetězce a získání informace slouží metoda *BarrierDetected*. Ta nemá žádné vstupní parametry, vrací pravdivostní hodnotu *true/false* v závislosti na tom, jestli je před robotem překážka.

Z Arduina je po částech přenesen řetězec obsahující vzdálenost z ultrazvukového měřiče vzdálenosti v cm a součet hodnot z infračervených senzorů. Infračervený senzor vrací hodnotu 1 pokud je před ním překážka, jinak vrací 0, takže jednou z podmínek pro detekci překážky je, že součet hodnot z infračerveného senzoru nesmí být větší než nula. Podmínka pro vzdálenost je proměnná a dá se nastavit změnou hodnoty proměnné *min_distance*.

Při testování často docházelo k situaci, že řetězec z Arduina byl nekompletní, prázdný nebo zdvojený. Z tohoto důvodu je prováděno zpracování řetězce pouze pokud je jeho velikost mezi nulou a osmi bajty, v ostatních případech je řetězec ignorován.

```
bool BarrierDetected();
```

Další definovanou metodou pro čtení informací ze senzorů je *GetNorthIndex*. Funkce čte data z magnetometru, které následně převádí na úhel určující směr jihu a spočítá index LED diody, která je nejblíže směru severu. Výpočetní postup je následující:

1. Senzor magnetometru vrací dvě souřadnice vektoru určujícího směr jihu,
2. Pomocí funkce arcus tangens získáme z daných souřadnic vektoru úhel v intervalu $<-\pi;\pi>$,
3. Tento úhel je následně konvertován z radiánů do stupňů,
4. Nakonec je výsledek normalizován do intervalu 0° až 359°.

Naměřené hodnoty z magnetometru jsou ovlivněny okolními vlivy. Proto před samotným výpočtem je provedena kalibrace senzoru. Pro úspěšnou kalibraci senzoru je zapotřebí otáčet magnetometrem různými směry po dobu pár sekund. Během toho se měří maximální a minimální hodnoty vektoru určujícího směr. Z naměřeného maxima a minima jednotlivých složek vektoru se spočítá průměr. Při měření je pak použit tento průměr k „posunutí“ vektoru. Tato kalibrace je provedena na začátku běhu programu metodou *ChassisIntro*.

```
int GetNorthIndex(CompassData &comp);
```

### 8.4.7 Hlavní programová smyčka

S využitím výše popsaných metod dokáže robot zpracovávat a prezentovat informace z okolního světa. V souboru *main.cpp* je metoda *main*, která je spouštěna při startu programu a definuje chování robota.

Prvním krokem je počáteční inicializace zahrnující spoustu kroků jako vynulování kalibračních proměnných, zavolání inicializačních metod jednotlivých komponent, nastavení detekčních slov a jejich citlivosti, kalibrace kompasu atd. Po inicializaci se vstoupí do hlavní smyčky programu, která je řízena návratovou hodnotou metody *ListenToHotword* detekující zvolání klíčového slova.

Návratová hodnota je struktura *HotwordInfo* obsahující index hlasového povelu a v případě, že se jedná o slovo „R2D2“, tak metoda rovněž vrací směr zvuku. Reakce na jednotlivé povely jsou následující

- *R2D2* – Na základě směru jsou rozsvíceny zelené LED diody metodou *LedsByMicIndex*. Následně je volána metoda *TurnToFace*, která zajistí aby se robot otočil čelem k osobě, která mluvila.
- *Forward* – Zavoláním metody *MoveChassis* s parametrem *C_FORWARD* se robot bude pohybovat směrem dopředu. Zastaví se pokud dojde k jedné z následujících podmínek

  - Vyslovení příkazu „Stop“,
  - Detekce překážky jedním z infračervených senzorů,
  - Vzdálenost mezi ultrazvukovým senzorem a překážkou je menší než nastavená hodnota (výchozí je 20cm),

- *Camera* – Robot spustí detekci pohlaví osob metodou *GenderFaceCount*. Mezitím běží odpočet na LED diodách, délka odpočtu je vstupním parametrem metody,
- *Left*/*Right*/*Backward*/*Stop* – pokyny pro pohyb robota spuštěné pomocí metody MoveChassis s parametry *C_LEFT*, *C_RIGHT*, *C_BACKWARD* a *C_STOP* pro zastavení.

Po zpracování příkazu se dokončí iterace cyklu a metoda se zase vrací k čekání na povel v metodě *ListenToHotword*. Předpokladem k naslouchání všem klíčovým slovům (kromě „R2D2“) je nejprve zavolání budícího slova „R2D2“ a následné vyslovení povelu v časovém intervalu indikovaném na LED diodách metodou *LedsCircle*.

```
#define C_STOP 0
#define C_LEFT 1
#define C_RIGHT 2
#define C_FORWARD 3
#define C_BACKWARD 4
```

# 9 Verifikace navrženého řešení

Poslední částí práce je otestování popsaných metod pro zpracování obrazu a zvuku. Měření úspěšnosti metod pro analýzu obrazu je testováno v rámci uložených obrázků bez proměnného vstupu kamery, aby bylo měření co nejpřesnější. Toto není nutné provést u testování zvuku, protože se neprovádí porovnání mezi metodami, ale pouze se vyhodnotí úspěšnost detekce.

## 9.1 Rozpoznání pohlaví

Testování zahrnuje použití tří testovacích databází, a to AT&T Facedatabase, Caltech face database a BioID face database. Testovací projekt se jmenuje *testovaniTvari*, je uložen mimo hlavní projekt, který ovládá robota a je součástí přílohy této práce. Testování je uloženo zvlášť, protože neprobíhalo na Raspberry Pi, ale na mém PC s Windows 10 z důvodu vyššího výkonu. Tento PC je vybaven procesorem Intel Core i7-3540M s taktem 3GHz a má k dispozici operační paměť o velikosti 8GB.

Projekt *testovaniTvari* obsahuje několik metod

- *FaceCrop* – Metoda pro zadané obrázky umožňuje s využitím klasifikátoru Viola-Jones výřez obličejů z obrázku a opětovné uložení samotného obrázku do specifikované složky,
- *FaceModelTraining* – Metoda slouží k natrénování a vytvoření klasifikátorů Eigenfaces, Fisherfaces a LBPH. Vstupním parametrem metody je cesta k CSV souboru se vstupními daty, cesta k výstupnímu souboru s natrénovaným modelem a druh klasifikátoru,
- *FaceModelTesting* – Cílem metody je otestovat klasifikátor vytvořený metodou FaceModelTraining a vytisknout do konzole základní údaje o výsledcích testování. Vstupním parametrem je cesta k modelu, CSV soubor s testovacími daty, druh klasifikátoru a rozlišení testovacích dat pro případnou změnu velikosti (testovací a trénovací obrázky musí mít stejnou velikost).

V main funkci projektu je volání jednotlivých metod. Výstupem trénování klasifikátorů je XML soubor s údaji pro klasifikaci. Doba trénování a velikost XML souboru výsledného klasifikátoru je rozdílná pro jednotlivé metody v závislosti na velikosti vstupních dat podle tabulky 4.

Z tabulky jsou zřejmé rozdíly při trénování jednotlivých metod. Velmi rozdílná je velikost modelu vytvořeného klasifikátorem Fisherfaces oproti ostatním metodám, zejména model metody Eigenfaces, který je v průměru ~37,5x větší při zachování relativně srovnatelného času pro jeho vytvoření. U klasifikátoru LBPH je zřejmé, že velikost vstupních dat neovlivňuje v takové míře dobu trénování ani výslednou velikost modelu.

Doba trénování a velikost klasifikátorů sice není důležitou metrikou při rozpoznání obličeje, ale může ovlivnit možnosti použití na zařízeních s omezenými výpočetními zdroji. Hlavní metrikou pořád zůstává úspěšnost detekce. Pro její testování jsou ve všech třech datasetech vytvořeny

Tabulka 4: Porovnání velikostí modelů a rychlosti trénování pro jednotlivé metody a databáze

| | Eigenfaces | | Fisherfaces | | LBPH | |
|---|---|---|---|---|---|---|
| | Čas[s] | Velikost[kB] | Čas[s] | Velikost[kB] | Čas[s] | Velikost[kB] |
| AT&T | 30 | 101676 | 19 | 11293 | 45 | 36088 |
| Caltech | 39 | 108741 | 24 | 583 | 53 | 41485 |
| BioID | 278 | 259842 | 262 | 660 | 90 | 89583 |

Tabulka 5: Porovnání úspěšnosti a rychlosti detekce pohlaví testovacích dat různých databází metodou Eigenfaces

| **Eigenfaces** | | AT&T | Caltech | BioID |
|---|---|---|---|---|
| AT&T | úspěšnost<br>čas | 97.5 %<br>21.075 ms | 61.54%<br>23.58 ms | 64 %<br>21 ms |
| Caltech | úspěšnost<br>čas | 42.5 %<br>11 ms | 100 %<br>10.5 ms | 73 %<br>10.46 ms |
| BioID | úspěšnost<br>čas | 55 %<br>25.35 ms | 65.38 %<br>28.04 ms | 100 %<br>24.18 ms |

testovací a trénovací data, kdy testovací data představují ~10% vstupních dat a zbylých 90% je určeno pro trénování. Seznam testovacích a trénovacích obrázků je uložen v CSV souborech v jednotlivých složkách datasetů s názvy *out_test.csv* resp. *out_train.csv*. Každý řádek v CSV souboru obsahuje cestu k souboru na disku a kategorii určující pohlaví (1 – muž, 100 – žena). V tabulkách 5, 6 a 7 jsou statistiky vyplývající z testování klasifikátoru na datech ze stejného datasetu a na datech ostatních datasetů.

Úspěšnost při testování jednotlivých metod na vlastních datech, tedy když jsou trénovací a testovací data ze stejného datasetu, dosahuje vysokých hodnot přes 90%. Na druhou stranu při

Tabulka 6: Porovnání úspěšnosti a rychlosti detekce pohlaví testovacích dat různých databází metodou Fisherfaces

| **Fisherfaces** | | AT&T | Caltech | BioID |
|---|---|---|---|---|
| AT&T | úspěšnost<br>čas | 100 %<br>1.6 ms | 61.54 %<br>1.27 ms | 54 %<br>1.43 ms |
| Caltech | úspěšnost<br>čas | 77.5 %<br>0.2 ms | 100 %<br>0.57 ms | 70 %<br>0.18 ms |
| BioID | úspěšnost<br>čas | 67.5 %<br>1.425 ms | 50 %<br>1.54 ms | 100 %<br>1.37 ms |

Tabulka 7: Porovnání úspěšnosti a rychlosti detekce pohlaví testovacích dat různých databází metodou Local Binary Patterns Histogram

| **LBPH** | | AT&T | Caltech | BioID |
|---|---|---|---|---|
| AT&T | úspěšnost<br>čas | 97.5 %<br>126.23 ms | 61.54 %<br>124.69 ms | 66 %<br>124.94 ms |
| Caltech | úspěšnost<br>čas | 82.5 %<br>128.68 ms | 96.15 %<br>132.42 ms | 68 %<br>129.66 ms |
| BioID | úspěšnost<br>čas | 75 %<br>205.8 ms | 65.38 %<br>204.85 ms | 100 %<br>201.62 ms |

použití testovacích a trénovacích dat z různých datasetů je úspěšnost mnohem menší. Pro jednotlivé datasety i metody jsou průměrné hodnoty podobné a pohybují se okolo ~70%. Z tohoto důvodu je jasné, že klasifikátory jsou úspěšnější, pokud testovaný obličej je rovněž zahrnutý v trénovací sadě. Lze tedy předpokládat, že zvýšením počtu trénovacích vzorků lze dosáhnout vyšší přesnosti detektoru, protože by byl složen z tváří, které se můžou podobností více blížit těm trénovacím. Dalším údajem patrným z výsledků testování je délka trvání klasifikace. V rámci tohoto faktoru je nejlepší metoda Fisherfaces, která dokáže provést rozpoznání průměrně za 1ms. Metoda Eigenfaces dosahuje přijatelné hodnoty okolo ~18ms, nejhorší je pak výsledek metody LBPH, která potřebuje k detekci průměrně ~155ms. Na základě výsledků měření byl vybrán klasifikátor jako nejvýhodnější klasifikátor Fisherfaces, díky jeho rychlosti a malé paměťové náročnosti při zachování podobné úspěšnosti jaké dosahují ostatní klasifikátory. Při použití rozsáhlejší trénovací sady by mohl klasifikátor dosahovat lepší přesnosti a úspěšnosti detekce.

## 9.2 Určení směru zvuku a rozpoznání klíčového slova

Testování zvukové analýzy bylo prováděno voláním klíčových slov na robota a zaznamenáváním správných a špatných detekcí slova z různých vzdáleností. Zároveň bylo ověřeno i správné určení směru zvuku. Každé měření zahrnuje pět pokusů vyslovení klíčového slova při jedné pozici (otočení) robota vzhledem k osobě provádějící testování. Toto testování probíhalo pro klíčové slovo „R2D2“, protože je to nejdůležitější slovo při životním cyklu robota. Úhly natočení robota odpovídají pozicím mikrofonů vzhledem k čelní straně robota. Z tohoto důvodu nezačíná první hodnota na 0°. Za správný výsledek detekce se považuje situace, kdy robot správně vyhodnotí klíčové slovo i směr odkud přišlo. Celkový počet měření je 160 pokusů, jejichž výsledky jsou uvedeny v tabulce 8.

Z tabulky 8 lze vyčíst, že testování je celkem přesné (83,75%), když se na robota mluví z malé vzdálenosti (1 metr). Se stoupající vzdáleností úspěšnost detekce výrazně klesá, obzvlášť schopnost detekovat směr. Pro vzdálenost tří metrů bylo provedeno jen několik měření, při kterých bylo zjištěno, že schopnost detekce je negativně ovlivněna okolními objekty a výsledky jsou

Tabulka 8: Procentuální úspěšnost správné detekce směru a klíčového slova z různých vzdáleností

| Směr<br>Vzdálenost | +22,5° | +67,5° | +112,5° | +157,5° | +202,5° | +247,5° | +292,5° | +337,5° |
|---|---|---|---|---|---|---|---|---|
| 1m | 90% | 90% | 100% | 70% | 70% | 80% | 70% | 100% |
| 2m | 30% | 50% | 30% | 20% | 20% | 30% | 30% | 50% |

Tabulka 9: Úspěšnost rozpoznání detekovaných slov a odolnosti vůči náhodným slovům s použitím mého hlasu

| | 1.kolo | 2.kolo | 3.kolo | 4.kolo | 5.kolo |
|---|---|---|---|---|---|
| Detekovaná slova | 85.71% | 71.42% | 85.71% | 85.71% | 71.42% |
| Náhodná slova | 100% | 100% | 100% | 100% | 100% |

zkreslené. Další měření bylo prováděno postupným vyslovováním sedmi anglických slov, která jsou zahrnutá v detekci („R2D2“, „forward“, „turn left“, „turn right“, „backward“, „stop“ a „camera“) a dalších sedmi anglických slov („hello“, „Artur“, „pop“, „caramel“, „bright“, „theft“, „robot“) z nichž jsou některá podobná detekovaným slovům a jiná čistě náhodně vybraná. Testování probíhalo vyslovením těchto čtrnácti slov postupně ze vzdálenosti jednoho metru. Tento postup se opakoval pětkrát a výsledky testování jsou uvedeny v tabulce 9. Hodnoty v tabulce představují procentuální úspěšnost detekce resp. nedetekce pro očekávaná a náhodná slova.

Z výsledků v tabulce 9 vyplývá, že detektor je velmi odolný vůči neočekávaným slovům. Úspěšnost detekce očekávaných slov je o něco nižší a dosahuje v průměru ~79,99%. Vzhledem k tomu, že klíčová slova jsou natrénována na hlas autora bylo zapotřebí otestovat detektor na cizí hlas. Konkrétně byl použit umělý hlas generovaný pomocí TTS (text-to-speech) enginu. [37] Další test tedy probíhal stejně jako ten předchozí pouze s použitím online TTS enginu s využitím mužského hlasu. Zvuk byl pouštěn skrz reproduktory, přičemž vzdálenost mezi reproduktory a robotem byla dvacet centimetrů. Výsledky testování s využitím umělého hlasu TTS enginu jsou v tabulce 10.

Při použití TTS enginu se úspěšnost detekce očekávaných slov dramaticky zhoršila. Může za to především rozdíl ve výslovnosti slov (důraz na slabiky apod.). Naopak odolnost detektoru proti náhodným slovům zůstala velmi silná viz tabulka 10.

Tabulka 10: Úspěšnost rozpoznání detekovaných slov a odolnosti vůči náhodným slovům s použitím hlasu z TTS enginu

| | 1.kolo | 2.kolo | 3.kolo | 4.kolo | 5.kolo |
|---|---|---|---|---|---|
| Detekovaná slova | 57.14% | 57.14% | 85.71% | 42.86% | 42.86% |
| Náhodná slova | 85.71% | 100% | 100% | 100% | 100% |

# 10 Závěr

Cílem práce bylo seznámení se s existujícími metodami pro analýzu zvuku a obrazu a navržení využití vybraných metod. Po prostudování metod bylo implementováno řešení využívající tyto metody v reálném světě. Sekundárním cílem práce byla realizace robota, který umožňuje ověření implementovaných metod v praxi.

V rámci analýzy zvuku byla teoretická část práce zaměřena na detekci směru (umístění zdroje) zvukových vln pomocí kruhového mikrofonového pole složeného z MEMS mikrofonů. S využitím korelace signálů a Fourierovy transformace bylo vytvořeno řešení, které podle výsledků testování je velmi dobré na krátké vzdálenosti do jednoho metru. Na delší vzdálenosti nedosahuje řešení takové přesnosti, což může být ovlivněno jak podmínkami při testování, tak kvalitou zpracování MEMS mikrofonů. Kromě určení směru zvukového signálu bylo navíc využito existujícího softwarového řešení Snowboy pro detekci klíčového slova v datovém proudu digitalizovaného zvukového signálu. Úspěšnost správné detekce byla rovněž nepřímo úměrná vzdálenosti mezi zdrojem zvuku a mikrofonovým polem, kdy nejlepších výsledků se dosáhlo při krátké vzdálenosti. S narůstající vzdáleností mezi robotem a obsluhující osobou klesala procentuální korektnost výsledků obou dílčích úkonů. Výslednou kombinací obou metod vzniklo řešení, díky němuž je robot schopen reagovat na povely vyslovené uživatelem.

Druhou teoretickou částí byla analýza obrazu, ve které se práce zaměřuje na detekci obličeje v obraze a rozpoznání pohlaví z tohoto obličeje. Pro detekci obličeje je využita metoda detekce Haarových příznaků v kombinaci s metodou strojového učení AdaBoost a kaskádou klasifikátorů pro rychlejší filtraci výsledků. Detekce obličeje v obraze dosahuje v tomto řešení dobrých výsledků a díky využití kaskády klasifikátorů je i dostatečně rychlá pro zpracování živého výstupu z kamery. Pro rozpoznání pohlaví z obličeje osoby jsou popsány tři metody – Eigenfaces, Fisherfaces a Local Binary Pattern Histogram. Z výsledků testování byla pro rozpoznání pohlaví využita metoda Fisherfaces. Ta sice dosahovala podobných hodnot správné detekce jako ostatní metody, ale dokázala detekci provést v rychlejším čase při využití menšího paměťového prostoru. Všechny metody dosahovaly průměrné úspěšnosti detekce okolo 70 % při simulaci reálné situace, tedy testování na osobách, které nebyly obsaženy v trénovací sadě. Výsledná míra úspěšnosti rozpoznání pohlaví je pouze o 20 % lepší než náhodný odhad, což je do jisté míry způsobeno nedostatkem trénovacích dat a je to zde prostor pro zlepšení.

Kromě implementace vybraných metod byla praktická část práce věnována realizaci robota. Z mechanického pohledu byly pro základ robota použity plastové desky, spojovací materiál, všesměrová transportní kolečka a motory s dalšími dvěmi kolečky zajišťující pohyb. Z pohledu využité výpočetní techniky byl robot vybaven počítačem Raspberry Pi, na kterém je spuštěn program zajišťující ovládání chování robota. Ten je doplněn mikropočítačem Arduino Nano, který zprostředkovává data z ultrazvukového měřiče vzdálenosti a infračervených detektorů překážek. Dále je k hlavnímu PC připojena kamera sloužící k získání vstupních dat pro popsanou metodu rozpoznání pohlaví. Pro interakci s uživatelem a zpracování zvukových signálů jsou

využity moduly na desce Matrix Creator.

S využitím popsaných součástí a implementovaných metod je robot schopen poslouchat příkazy uživatele a reagovat na ně pohybem a signalizací na LED diodách. Z důvodu celkové velikosti robota a vlastností jednotlivých součástí je možné se s robotem pohybovat pouze po rovném povrchu. Pro zachování maximální mobility robot nepotřebuje k provozu internetové připojení a použitá powerbanka jej dokáže udržet v chodu několik hodin.

# Literatura


[1] Zpracování řečových signálů - studijní opora. Fakulta informačních technologií VUT v Brně [online]. Brno: doc. Černocký, Speech@FIT, 2006 [cit. 2019-04-23]. Dostupné z: http://www.fit.vutbr.cz/study/courses/ZRE/public/opora/zre_opora.pdf

[2] MILLER Frederic P., VADOME Agnes F.a McBREWSTER John. Nyquist-Shannon Sampling Theorem. Alphascript Publishing, 1. vydání, 2010. ISBN 978-613-0-04581-4.

[3] MIŠUREC, Jiří. Základní metody číslicového zpracování signálů pro integrovanou výuku VUT a VŠB-TUO [online]. 1. Vysoké učení technické vBrně, 2014 [cit. 2019-04-26]. ISBN 978-80-214-5118-6. Dostupné z: https://vut-vsb.cz/home/get-file?file=465

[4] Radioelektronická měření (MREM, LREM) - A/D a D/A převodníky. Ústav radioelektroniky FEKT VUT v Brně [online]. Brno: Ing. Dřínovský, 2010 [cit. 2019-04-23]. Dostupné z: www.urel.feec.vutbr.cz/~drinovsky/?download=MREM_2010_P_02.pdf

[5] KUČERA, Antonín. Zpracování a analýza zvukového signálu na robotickém zařízení [online]. Ostrava, 2014 [cit. 2019-04-24]. Dostupné z: http://hdl.handle.net/10084/103788. Diplomová práce. Vysoká škola báňská - Technická univerzita Ostrava

[6] But what is the Fourier Transform? A visual introduction. In: Youtube [online]. 3Blue1Brown, 2018 [cit. 2019-04-24]. Dostupné z: https://youtu.be/spUNpyF58BY

[7] FFT: An 8 input butterfly [online]. AlwaysLearn, 2019 [cit. 2019-04-24]. Dostupné z: http://www.alwayslearn.com/DFT%20and%20FFT%20Tutorial/DFTandFFT_FFT_Butterfly_8_Input.html

[8] Align Signals Using Cross-Correlation - MATLAB & Simulink [online]. Mathworks [cit. 2019-04-24]. Dostupné z: https://www.mathworks.com/help/signal/ug/align-signals-using-cross-correlation.html

[9] HÁJOVSKÝ, Radovan, Radka PUSTKOVÁ a František KUTÁLEK. Zpracování obrazu v měřící a řídící technice: učební text : studijní materiály pro studijní obor Měřicí a řídicí technika, Elektronika Fakulty elektrotechniky a informatiky [online]. Ostrava: Vysoká škola báňská - Technická univerzita, 2012 [cit. 2019-04-26]. ISBN 978-80-248-2596-0. Dostupné z: http://www.person.vsb.cz/archivcd/FEI/ZOMRT/

[10] VIOLA, Paul a Michael JONES. Robust Real-time Object Detection [online]. Vancouver, Kanada, 2014 [cit. 2019-04-24].
Dostupné z: http://www.face-rec.org/algorithms/Boosting-Ensemble/16981346.pdf

[11] V. HOANG, A. VAVILIN a K. JO. Pedestrian detection approach based on modified Haar-like features and AdaBoost [online]. JeJu Island, Jižní Korea, 2012 [cit. 2019-04-24]. Dostupné z:
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6393256&isnumber=6393028

[12] BARTOŠ, Václav. Detekce protijedoucích vozidel z obrazů za pomocí kamery [online]. Ostrava, 2016 [cit. 2019-04-26]. Dostupné z: http://hdl.handle.net/10084/116075. Bakalářská práce. Vysoká škola báňská - Technická univerzita Ostrava.

[13] Face Recognition Methods & Applications. Divyarajsinh N Parmar et al: Int.J.Computer Technology & Applications [online]. 2009(4), 84-86 [cit. 2019-04-26]. ISSN 2229-6093. Dostupné z: https://arxiv.org/ftp/arxiv/papers/1403/1403.0485.pdf

[14] R. BRUNELLI a T. POGGIO. Face recognition through geometrical features. [online]. Trento, Itálie, 1992 [cit. 2019-04-24]. Dostupné z: https://tev-static.fbk.eu/people/brunelli/Papers/EC.pdf

[15] BELHUMEUR, Peter N., HESPANHA, João P. a KRIEGMAN, David J., Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern Analysis & Machine Intelligence [online] 1997, 7: 711-720.
Dostupné z: https://www.computer.org/csdl/trans/tp/1997/07/i0711.pdf

[16] SMITH, Lindsay I. A tutorial on principal components analysis. [online]. 2002 [cit. 2019-04-24]. Dostupné z: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

[17] AHONEN, Timo; HADID, Abdenour; PIETIKÄINEN, Matti. Face recognition with local binary patterns. [online]. In: European conference on computer vision. Springer, Berlin, Heidelberg, 2004, 469-481. [cit. 2019-04-24]. Dostupné z: https://link.springer.com/chapter/10.1007/978-3-540-24670-1_36

[18] SADEGH M.SC, Reihaneh. Researches: Facial expression recognition using appearance-based statistical method. In:Reihaneh Sadegh, M.SC: Yazd University[online]. 2014 [cit. 2019-04-24]. Dostupné z: https://pws.yazd.ac.ir/rezaeian/Sadegh/researches.html

[19] Face Recognition with OpenCV [online] OpenCV [cit. 2019-04-24]. Dostupné z: https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html

[20] Face recognition homepage [online]. prof. Mislav GRCIC, Ph.D. a Kresimir DELAC, Ph.D., 2007 [cit. 2019-04-23]. Dostupné z: http://face-rec.org/databases/

[21] The Database of Faces [online]. AT&T Laboratories Cambridge, 1994 [cit. 2019-04-23]. Dostupné z: https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[22] Frontal face dataset [online]. California Institute of Technology, 1999 [cit. 2019-04-23]. Dostupné z: http://www.vision.caltech.edu/html-files/archive.html

[23] The BioID face database [online]. BioID, Německo [cit. 2019-04-23]. Dostupné z: https://www.bioid.com/facedb

[24] Yale Face Database [online]. Computer Science @ Yale University, 1997 [cit. 2019-04-23]. Dostupné z: http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html

[25] Raspberry Pi GPIO. In: Big Mess o' Wires [online]. Steve Chamberlin, 2018 [cit. 2019-04-23]. Dostupné z: https://www.bigmessowires.com/2018/05/26/raspberry-pi-gpio-programming-in-c/

[26] Raspberry Pi Documentation [online]. Raspberry Pi Foundation, 2019 [cit. 2019-04-23]. Dostupné z: https://www.raspberrypi.org/documentation/

[27] Arduino - introduction [online]. Arduino, 2019 [cit. 2019-04-23]. Dostupné z: https://www.arduino.cc/en/Guide/Introduction

[28] Overview MATRIX [online]. Matrix Labs, 2019 [cit. 2019-04-23]. Dostupné z: https://matrix-io.github.io/matrix-documentation/

[29] Matrix Creator datasheet [online]. Matrix Labs, 2017 [cit. 2019-04-23]. Dostupné z: http://www.farnell.com/datasheets/2357255.pdf

[30] Arduino ultrasonic sensor (HC-SR04 or HY-SRF05) [online]. Energia Zero [cit. 2019-04-23]. Dostupné z: http://www.energiazero.org/arduino_sensori/Arduino%20ultrasonic%20sensor%20(HC-SR04%20or%20HY-SRF05).pdf

[31] SRF05 - Ultra-Sonic Ranger [online]. Robot Electronics [cit. 2019-04-23]. Dostupné z: https://www.robot-electronics.co.uk/htm/srf05tech.htm

[32] Infračervený optický senzor [online]. Arduino návody, 2016 [cit. 2019-04-23]. Dostupné z: https://navody.arduino-shop.cz/navody-k-produktum/infracerveny-opticky-senzor.html

[33] Snowboy, a Customizable Hotword Detection Engine [online]. KITT.AI, 2017 [cit. 2019-04-23]. Dostupné z: http://docs.kitt.ai/snowboy/

[34] Advanced Linux Sound Architecture (ALSA) project homepage [online]. AlsaProject, 2019 [cit. 2019-04-23]. Dostupné z: https://www.alsa-project.org/wiki/Main_Page

[35] OpenCV 3.4.0 [online]. OpenCV, 2017 [cit. 2019-04-23]. Dostupné z: https://docs.opencv.org/3.4.0/

[36] RS-232 for Linux, FreeBSD and windows [online]. Teunis van Beelen, 2017 [cit. 2019-04-23]. Dostupné z: https://www.teuniz.net/RS-232/

[37] From text to speech [online]. [cit. 2019-04-23]. Dostupné z: http://www.fromtexttospeech.com/

# A Přílohy v IS Edison

- Implementace projektu uložena v adresáři */implementace*
- Implementace jednotlivých testovacích projektů uložené v adresáři */testovaci_projekty*
- Konfigurace ALSA softwaru uložena v adresáři */konfigurace*
- Nahrávky zvukových povelů ve formátech WAV a PDML v adresáři */zvukove_modely*